University of Glasgow
DEPARTMENT OF
AEROSPACE
ENGINEERING

Non-linear Analysis of Stall Flutter Based on the ONERA Aerodynamic Model

Aerospace Engineering Report 0205b

Jeremy Beedy and George Barakos

Non-linear Analysis of Stall Flutter Based on the ONERA Aerodynamic Model

Aerospace Engineering Report 0205b

Jeremy Beedy and George Barakos

Department of Aerospace Engineering

University *of* Glasgow

Glasgow G12 8QQ

United Kingdom

July 2002

Contents

List of Figures

Nomenclature

Symbol	Definition
a_{0z}	Linear slope of "general" aerodynamic force coefficient
a_{0L}, a_{0M}	Slopes of linear coefficient curves -lift and moment.
b	Semi-chord length $b = \frac{c}{2}$
b_1	Non-linear slope of the deviation from linear force curve
c	Chord length
C(k)	Theodorsen function
C_z	Total aerodynamic force coefficient (z=L for lift, z=M for moment)
C_{z1}	Linear contribution to the "general" aerodynamic force coefficient
C_{z2}	Non-linear contribution to the "general" aerodynamic force coefficient
$C_{z\gamma}$	Linear circulatory contribution to the "general" aerodynamic force coefficient
C_{z0}	Value of the "general" aerodynamic force coefficient
C_{zs1}	First harmonic sine component of the "general" aerodynamic force coefficient
C_{zc1}	First harmonic cosine component of the "general" aerodynamic force coefficient
C_{zs2}	Second harmonic sine component of the "general" aerodynamic force coefficient
C_{zc2}	Second harmonic cosine component of the "general" aerodynamic force coefficient
C_{z10}	Mean component of the linear contribution to the "general" aerodynamic force coefficient
C_{z1s}	Sine component of the linear contribution to the "general" aerodynamic force coefficient
C_{z1c}	Cosine component of the linear contribution to the "general" aerodynamic force coefficient

C_{z20}	Mean harmonic component of the non-linear contribution to the "general" aerodynamic force coefficient
C_{z2s1}	Sine first harmonic component of the non-linear contribution to the "general" aerodynamic force coefficient
C_{z2c1}	Cosine first harmonic component of the non-linear contribution to the "general" aerodynamic force coefficient
C_{z2s2}	Sine second harmonic component of the non-linear contribution to the "general" aerodynamic force coefficient
C_{z2c2}	Cosine second harmonic component of the non-linear contribution to the "general" aerodynamic force coefficient
$C_{z\dot{\gamma}0}$	Mean component of the linear circulatory contribution to the "general" aerodynamic force coefficient
$C_{z\dot{\gamma}s}$	Sine component of the linear circulatory contribution to the "general" aerodynamic force coefficient
$C_{z\dot{\gamma}c}$	Cosine component of the linear circulatory contribution to the "general" aerodynamic force coefficient
e	Distance from elastic axis to mid-chord
	Total energy of the fluid per unit volume
F(k)	Real part of the Theodorsen function
F_{Kleb}	Klebanoff's intermittency function
G(k)	Imaginary part of the Theodorsen function
h	Tip deflection about ¼ chord
h_0	Mean component of the ¼ chord deflection
h_s	Sine component of the ¼ chord deflection
h_c	Cosine component of the ¼ chord deflection
H_v	Source term associated with turbulence model
i	Internal energy
I_a	Mass moment of inertia per unit length
k	Reduced frequency $k = \frac{\omega \cdot b}{U}$
k_a	Torsional reduced frequency

K_a	Torsional stiffness per unit length
K_h	Bending stiffness per unit length
[K]	Stiffness matrix
k_{ij}	Components of the stiffness matrix
l	Wing length
l_{mix}	Mixing length
L	Lift
M	Total mass per unit length
M_a	Pitching moment
[M]	Mass matrix
M_{ij}	Components of the mass matrix
n	Number of modal shapes
p	Pressure
q_i	i-th modal amplitude
q_{io}	Mean component of the i-th modal amplitude
q_{is}	Sine component of the i-th modal amplitude
q_{ic}	Cosine component of the i-th modal amplitude
q	Dynamic pressure
$\dot{q}$	Heat flux
Q_i	i-th modal force
Q_{i0}	Mean component of the i-th modal force
Q_{is}	Sine component of the i-th modal force
Q_{ic}	Cosine component of the i-th modal force
r_0	Gyration radius
r_1, r_2, r_3	Coefficients of ONERA non-linear aerodynamic differential equations
r_{10}, r_{12}, r_{20}, r_{22}, r_{30}, r_{32}	Parabolic coefficients of ONERA non-linear aerodynamic equations
S	Wing surface area
S_{z1}, S_{z2}, S_{z3}	Coefficients of linear ONERA aerodynamic modes
S_a	Static moment of inertia per unit length
t	Real time

U	Free stream velocity
U_{dif}	Maximum value of U.
w	Velocity component in the z-direction
Greek symbols	**Definition**
α	Angle of attack
α_0	Mean component of the angle of attack
α_s	Sine component of the angle of attack
α_c	Cosine component of the angle of attack
α_1	Stall angle
α_v	Vibration amplitude of the angle of attack
γ_i	Non-dimensional mode shape
δ_{ij}	Kronecker's delta
ΔC_z	Deviation of the non-linear aerodynamic force curve from the linear approximation
ΔC_{z0}, ΔC_{zs1}, ΔC_{zc2}	First and second harmonic oscillatory amplitudes of the deviation of the non-linear aerodynamic force curve from the linear approximation
ε_j	Parameter of the j-th beam bending mode shapes
$\Theta = \frac{\omega_h}{\omega_a}$	Ratio of bending to torsional frequency
θ	Pitch angle about ¼ chord
θ_0	Mean component of the pitch angle
θ_s	Sine component of the pitch angle
θ_c	Cosine component of the pitch angle
θ_R	Root angle of attack
κ	von Karman constant
λ_1, λ_2	Coefficients in linear aerodynamic force model
μ	Density ratio= $\frac{M}{\pi \cdot \rho \cdot b^2}$
ξ,ζ	Curvilinear coordinates
ρ	Free stream density

τ	Dimensionless time $= \frac{t \cdot U}{b}$
ϕ	Non-dimensional phase associated with the angle of attack
Φ_a	Beam torsion mode shape
Φ_h	Beam bending mode shape
ω	Real frequency
	Magnitude of the vorticity vector
ω_a	Uncoupled torsion frequency
ω_h	Uncoupled bending frequency
ω^*	Under-relaxation parameter

Chapter 1. Introduction

1.1. Background

A type of oscillation of airplane wings and control surfaces has been observed since the early days of flight. To describe the physical phenomenon, let us consider a cantilever wing, without sweepback and without aileron, mounted in a wind tunnel at a small angle of attack and with a rigidly supported root (Fig.1.1). When there is no flow in the wind tunnel, and the model is disturbed from its static position, oscillation sets in, which is gradually damped out. When the velocity of flow in the wind tunnel gradually increases, the rate of damping of the oscillation of the disturbed aerofoil first increases. With further increase of the flow velocity, however, a point is reached at which the damping rapidly decreases. At the critical *flutter velocity*, an oscillation can just maintain itself with steady amplitude. At velocities of the flow somewhat above the critical, a small disturbance of the aerofoil can serve as a trigger to initiate an oscillation of large amplitude. In such circumstances the aerofoil suffers from oscillatory instability and is said to flutter. From now on, the terms *flutter velocity* and *flutter frequency* refer to the critical flutter velocity and the frequency at the critical condition [1.1].

Experiments on wing flutter show that the oscillation is self-sustained, that is, no external oscillator or forcing agent is required. The motion can maintain itself or grow for a range of flow velocity, which is more or less wide according to the design of the wing and the conditions of the test. For a simple cantilever wing, flutter occurs at any flow velocity above the critical. For more complex structures, i.e. wings with ailerons, there may be one or more ranges of velocities for which flutter occurs, and these are bounded at both ends by critical velocities at which an oscillation of constant amplitude can just maintain itself.

It is well known that as an aerofoil with increasing angle of attack approaches a certain relatively large angle of attack (stall angle), lift begins to decrease rapidly and the so-called stall is reached (Fig.1.2). The aerodynamic mechanism of this stall is not explainable by potential flow theory any longer since separation of the flow from the aerofoil occurs. In some instances this separation occurs in an unstable way so that

fluctuating conditions ranging from near-potential flow conditions to very turbulent conditions are obtained. If a body is fluttering and, during part or all of the time of oscillation the flow is separated, then the flutter phenomenon is called a *stall flutter*.

Stall flutter is a serious aeroelastic instability for rotating machineries such as propellers, turbine blades, and compressors, which sometimes have to operate at angles of attack close to the static stalling angle of the blades. Airplane wings and tails rarely suffer from stall flutter. However, the trend toward thin wing sections and large wingspan has increasingly made stall flutter of wings a serious concern for the design of high-speed aircraft.

Most analysis of aircraft flutter behaviour is traditionally based on small amplitude, linear theory, particularly with respect to the aerodynamic modelling. However, if the wing is near the stall region, a non-linear stall flutter limit cycle may occur at a lower velocity than linear theory suggests. Since some current aircraft are achieving high angle of attack for manoeuvring, it is of interest to explore this non-linear stall flutter behaviour and its transition from linear behaviour.

The objective of the present work is to explore analytically the roles of non-linear aerodynamics in high angle-of-attack stall flutter while attempting to develop a simple non-linear method of flutter analysis.

The non-linear flutter calculations were carried out solving the Rayleigh-Ritz formulation using the ONERA model [1.2,1.3] as the basis for the aerodynamics. A harmonic balance method and a Newton-Raphson solver were applied to the resulting non-linear equations.

In order to fit some parameters of the ONERA model, a CFD code was used to calculate the aerodynamic coefficients of an aerofoil undergoing oscillatory motion. The code uses an implicit unfactored method with various turbulence models [1.4]. It combines Newton sub-iterations and point-by-point Gauss-Seidel sub-relaxations.

1.2. Literature survey

1.2.1. Aerolasticity

The potentially disastrous effects of *fluid-structure interaction* are not new problems in the field of aerospace engineering. The problem of wing divergence was apparent to early aircraft designers in that it caused a number of unexpected crashes. However, early attempts to cure the problem usually involved structural stiffening with little understanding of the aeroelastic nature of the problem. The development of aeroelasticity methods started in the early 1950s, but the scale of the problem is such that, after some 40 years of focused and innovative research, there are still no established numerical methods that can routinely predict flutter stability for representative geometries [1.5].

Broadly speaking, aerolasticity analysis can be divided into *classical* and *integrated methods*, the former ignoring the interaction between the fluid and the structure and the latter attempting to model it.

Classical aerolasticity methods are those where the fluid and structural domains remain uncoupled in such a way that the fluid flow does not affect the structural response, which is usually obtained from a rigid typical cross-section representation. Such methods thus split an inherently coupled non-linear phenomenon into two separate uncoupled analyses.

The aeroelastic eigensolution method has been by far the most popular of the classical flutter stability analyses. It is based on obtaining the linearized harmonic unsteady aerodynamic coefficients for the motion of a freely vibrating structure. The unsteady aerodynamics can be provided by anything from empirically determined section lift and moment coefficients to linearized potential methods. The structural model is often restricted to a two-degree-of-freedom typical section representation (see Fig.1.3), but the formulation is general, so that three-dimensional descriptions can also be accommodated. Once determined, the aeroelastic forces are expressed in the frequency domain, either directly if analytical theories are used, or by Fourier analysis if the forces were first calculated in the time domain. The resulting aeroelastic equations of motion are very

similar to the structural equations, the aerodynamic contribution being added to the mass and/or stiffness matrices. However, these new system matrices may well become a function of the frequency, in which case the eigenproblem is no longer mathematically linear. In such situations an approximate solution can be found using iterative techniques. One of the main disadvantages of the method lies in its simplified representation of the structural dynamics (usually a lumped parameter model), which allows parametric studies to be conducted with minimum computational effort.

Tran and Petot [1.2] and Dat and Tran [1.3] of the Office National d'Etudes et de Recherches Aérospatiales (ONERA) developed, in the 1980's, a semi-empirical, unsteady, non-linear model (called the ONERA model) for determining two-dimensional aerodynamic forces on an aerofoil oscillating in pitch only, which experiences dynamic stall. It was felt that this model, as amended for pitch and plunge by Peters [1.6] and by Petot and Dat [1.7], could provide a convenient means of including non-linear stalled aerodynamic forces into an analytical study of stall flutter.

Dunn and Dugundji [1.8] developed a simple analytic method to include non-linear structural and aerodynamic effects into a stall flutter analysis, using the ONERA model for the aerodynamics. Experimental data were obtained on a set of aeroelastically tailored wings with varying amounts of bending-torsion coupling. The analysis predicted reasonably almost all the observed, experimental, non-linear stalled phenomena on the wings tested.

The flutter behaviour of rotor blades has also been investigated. Chopra and Dugundji [1.9] gave an analysis of the non-linear aeroelastic behaviour and its transition from linear behaviour, but dealing only with geometrical non-linearities of the rigid blade.

Tang and Dowell [1.10] have introduced both structural non-linearities and dynamic stall in their investigation of stall limit cycles and chaotic motion of flexible, non rotating blades. In their work the structural non-linearities were approximated by the moderate deflection equations developed by Hodges and Dowell [1.11], and the dynamic stall was represented by the ONERA model. Kim and Dugundji [1.12] incorporated the structural effects into a non-linear, large amplitude flutter limit cycle analysis of rotating hingeless composite blades using the ONERA model.

1.2.2. CFD methods

With dramatic increases in computing power and advances in *computational fluid dynamics* (CFD) methods, it is now possible to obtain solutions for both inviscid (hyperbolic Euler equations) and viscous (parabolic Navier-Stokes equations) steady flows, including a reasonable representation of viscous losses. The usual approach is to use time integration, starting from known initial conditions. Unsteady flows are much more complex in nature because of temporal variations, but representative solutions can still be obtained if the time-dependent terms are discretized in a time-accurate fashion. Over the last two decades, time domain solutions have progressed through full-potential, linearized-Euler, Euler and, most recently, Reynolds-averaged Navier-Stokes equations.

Two distinct methods have been adopted to discretize the fluid domain. The first one, known as *finite-difference*, is based on the differential form of the governing equations. The physical grid on which the calculation is to be performed is transformed to a rectangular computational grid with equal spacings in each direction, onto which the equations are discretized. The second method, known as finite-volume, uses the integral form of the governing equations, which are discretized as fluxes through control volumes of known shape but arbitrary size. Further differences between the CFD methods lie in the integration schemes adopted (explicit or implicit) for solving the discretized equations (cell vertex or cell centred schemes), the meshing strategy used (structured- O, C, or H meshes; unstructured- adaptive, dynamic meshes) and the implementation of the boundary conditions (non-reflecting, periodic, moving boundaries)[1.5].

One of the major problems associated with three dimensional unsteady flow modelling is the amount of CPU time required to obtain fully converged solutions. The modelling of turbulence is probably one of the most significant challenges to the accuracy of modern CFD methods. For the foreseeable future, it is impractical to contemplate a direct application of the Navier-Stokes equations to turbulence as an unrealistically fine mesh would be required to capture the smallest eddies, an approximate size of which being provided by the Kolmogorov scale. Therefore, turbulence can only be represented using simplified models which are usually based on equivalent local turbulent viscosities given by the averaged effect of the eddies at some point in the flow. There is a hierarchy

of such methods which range from the fully empirical mixing length model (zero-equation eddy viscosity model: Baldwin & Lomax[1.13]), one-equation k-l models (Cebeci & Smith [1.14]; Johnson & King [1.15]; Baldwin & Barth [1.16]), two-equations k-ε models (Jones & Launder [1.17]), to the Reynolds stress model with six equations in 2D and nine equations in 3D.

1.2.3. Unsteady aerodynamics

Currently, the most accurate aerodynamic models are computational fluid dynamic solutions to the governing equations. These CFD-based approaches are often considered to be the salvation of the aeroelastician; however, in many situations, a time-marching aeroelastic analysis using a CFD method is very computationally expensive for routine design purposes or parametric studies.

There exists, therefore, a need to develop "rational" prediction methods so that expensive and time-consuming development testing can be minimized. There are broadly two different approaches available to achieve this goal, i.e. viscous-inviscid interaction methods and Navier-Stokes methods. The former method couples inviscid flows with boundary layer flows (Abdel-Rahim et al. [1.18], Jang et al. [1.19], Cebeci et al. [1.20]), whereas the latter relies on the solution of the full viscous flow equations.

Clarkson, Ekaterinaris and Platzer [1.21] attempted the analysis of aerofoil stall flutter by applying a Navier-Stokes flow solver to the problem of low subsonic flow, M=0.3, over an oscillating NACA-0012 aerofoil in the light stall regime. The flow was assumed fully turbulent, and the Baldwin-Lomax, the algebraic RNG-based model and the half-equation Johnson-King turbulence models were used. It was found that none of these models was capable of reproducing the hysteresis loops measured by McCroskey [1.22].

As is well known, modelling of the turbulence flow behaviour is an important issue in computational aerodynamics. The flow over an oscillating blade is often massively separated and involves multiple length scales. For the computation of these flows, application of algebraic turbulence models, such as the Cebeci-Smith, the Baldwin-Lomax, or the Johnson-King model, becomes very complicated and also

ambiguous. The source of this ambiguity comes from the difficulty in defining characteristic length scales, such as boundary layer thickness. An extensive investigation of the ability of these simple models as well as the recently developed Baldwin-Barth and Spalart-Almaras [1.23] one-equation models to predict unsteady separated flows has been conducted [1.24]. The one-equation models have shown superior behaviour compared to algebraic models.

Two-equation models, such as the k-ε and the k-ω models, have also been used to compute steady and unsteady aerofoil flows [1.25]. It appears that the k-ω model provides some improvements over the one-equation models. The k-ε model, on the other hand, does not predict well the adverse pressure gradient separated flow regions.

Numerical investigations [1.26] of aerofoil flows showed that the effect of the leading edge transitional flow region is of primary importance to the overall development of the suction side viscous flow region. Laminar/transitional separation bubbles form near the aerofoil leading edge for angles of incidence as low as 6 deg and significantly alter the suction side pressure distribution and the boundary layer formation.

Ekaterinaris and Platzer [1.27] show that it is important to take into account the leading edge transitional flow not only for the lower Reynolds number regime but also for the high Reynolds number in order to predict stall flutter. For the high Reynolds number regime, transition is expected to occur very close to the leading edge after the adverse pressure gradient region is encountered. The extent of the transition region and the separation bubble is also expected to be very small. Their results for an oscillating aerofoil show that modelling of the transitional flow near the leading edge decisively changes the character of the pitching moment hysteresis loop. Whereas the fully turbulent calculations produce only counter clockwise moment loops, the transitional calculations produce clockwise loops. Their finding is very important since it is well known (see [1.28]) that single-degree-of-freedom stall flutter in pitch occurs as soon as the area enclosed by the clockwise moment loop exceeds the area enclosed by the counter clockwise loop.

1.2.4. Dynamic Stall

The dynamic stall process has been under investigation for about three decades, and significant progress has been made towards understanding the physical processes associated with the rapidly pitching of an aerofoil beyond its static stall angle of attack.

In the 70's the study of unsteady turbulent flows was dominated by the work of Stuart, Telionis, Wirz, McCroskey and Shen. The above efforts are mainly experimental studies and attempts to derive analytical solutions for unsteady boundary layers. In the 80's the rapid progress of computer technology allowed researchers to simulate unsteady flows using numerical techniques. The reviews of McCroskey [1.29], Carr [1.30] and Visbal [1.31] provide good descriptions of the dynamic stall processes. In the 90's the effect of turbulence on dynamic stall has been the subject of extensive experimental [1.32] and numerical studies [1.33].

The earliest computational investigations of dynamic stall appeared in 70s and early 80s with indicative efforts by Mehta [1.34] and Gulcat [1.35]. In the middle 80s, the algorithm by Wu [1.36, 1.37] for incompressible flow provided results consistent with experimental data. Tuncer [1.38] extended the model for high Reynolds number flows, obtaining accurate and inexpensive results.

Compressibility effects have been addressed during the past few years, with high Reynolds number compressible flow solutions appearing infrequently. The time delay between the appearance of incompressible and compressible Navier-Stokes solutions primarily represents increased computational requirements. Compressibility adds an additional differential equation (for energy) to the system of equations to be solved. Furthermore, the solution must account for sharp flow-field gradients and include thermodynamic features such as shock waves.

High Reynolds number flows not only increase the computer demand but also require turbulence modelling. To date such modelling seems as much an art as a science. Examination of the available literature reveals that detailed investigations of dynamic stall including high Mach and Reynolds number effects as well as lower pitch rates and more complex aerofoils, are rare. High pitch rates generally produce more straight-forward, vortex-dominated flows where turbulence is less pronounced. Low pitch rate

flow fields are usually more difficult to compute. Finally, complex aerofoils require complex grid generators and exhibit edges, such as truncated trailing edges, which can produce numerical difficulties.

1.3. Objectives

The objective of the project is to develop a method of analysing flutter in aerofoils.

The flutter calculations are based on a non-linear model for the aerodynamic part (ONERA model) and on a linear model for the structural part. The non-linear equation of the ONERA model has several parameters, which have to be supplied to the model, for each case, from either:

i) experimental data, or

ii) CFD simulation.

Several cases are solved using the unsteady Parallel Multi-Block (PMB) CFD code to calculate the loads on an aerofoil under certain oscillating conditions, and hence obtaining the lift and moment loops. The solver combines Newton sub-iterations and point-by-point Gauss-Seidel sub-relaxation. For this study, a k-ω turbulence model is used. Some cases are solved as fully turbulent problems and others with tripping. The grid is not very fine, and the time resolution is the minimum possible, in order to simplify the calculations.

Using the results from the code, and the available experimental data, the parameters of the ONERA model are fitted. The fitting is done using the Levenberg-Marquardt method, a non-linear fitting method.

Once the parameters are fitted for every case, the flutter program is run to obtain the flutter velocity and frequency of the aerofoil.

1.4. Report outline

The report is structured in the following parts:

Chapter 2. Aeroelastic model: the ONERA model and the basic calculations for the flutter are explained here. This is the basis of the flutter program.

Chapter 3. Results: first, the solved cases are described. After that, the results are presented and discussed.

Chapter 4. Conclusions: the most relevant findings are briefly summarized and suggestions for the future are made.

Appendices Details of the computer implementation of the model along with description of the employed numerical methods are given in the appendices.

Chapter 2. Aeroelastic model

An aeroelastic model consists of two parts: aerodynamic and structural. Non-linearities in the structural part can arise from geometry and material while in the aerodynamic part they can arise from flow separation and stall. In the present work only aerodynamic non-linearities are considered.

2.1. Aerodynamic model

The ONERA aerodynamic model [2.1] was used in this study in order to solve the aerodynamic non-linearities. According to this model all airloads can be expressed as the sum of a linear and non-linear part as:

$$C_z = C_{z1} + C_{z2} \tag{2.1}$$

The linear part, C_{z1}, is given by:

$$C_{z1} = S_{z1} \cdot \dot{\alpha} + S_{z2} \cdot \ddot{\theta} + S_{z3} \cdot \dot{\theta} + C_{z\gamma} \tag{2.2}$$

$$\dot{C}_{z\gamma} + \lambda_1 \cdot C_{z\gamma} = \lambda_1 \cdot a_{0z} \cdot (\alpha + \dot{\theta}) + \lambda_2 \cdot a_{0z} \cdot (\dot{\alpha} + \ddot{\theta}) \tag{2.3}$$

While the non-linear one, C_{z2}, is:

$$\ddot{C}_{z2} + r_1 \cdot \dot{C}_{z2} + r_2 \cdot C_{z2} = -r_2 \cdot \Delta C_z\big|_\alpha - r_3 \cdot \Delta \dot{C}_z\big|_\alpha \tag{2.4}$$

In the above equations:

$\alpha = \theta - \dot{\tilde{h}}$ (Incidence angle, separated into pitching and plunging, see fig.2.1)

(2.5)

$$\left(\tilde{}\right) = \frac{\left(\right)}{b} \qquad \left(\dot{}\right) = \frac{\partial\left(\right)}{\partial\tau} \qquad \left(\ddot{}\right) = \frac{\partial^2\left(\right)}{\partial\tau^2} \qquad \tau = \frac{U \cdot t}{b} \tag{2.6}$$

$\left(\dot{}\right)$ Stands for the first time derivative, and $\left(\ddot{}\right)$ for the second time derivative. ΔC_z is the deviation of the non-linear aerodynamic curve from the linear approximation (Fig.2.2).

For this model, z represents a general aerodynamic load, which can be lift (L) or moment (M), and the parameters of the linear equations are:

S_{L1}=3.142 S_{L2}=1.571 S_{L3}=0.0

S_{M1}= -0.786 S_{M2}= -0.589 S_{M3}= -0.786

a_{0L}=6.28 a_{OM}=0.0

λ_1=0.15, λ_2=0.55 (same for lift and moment calculations)

where a_{0L} and a_{0M} represent the slopes of the linear coefficient curves.

The term C_{zK} (circulatory contribution to the linear aerodynamic coefficient) accounts for the aerodynamic lag due to the formation of flow structures, like vortices.

The total static lift or moment coefficient can be calculated as:

$$C_z(\alpha) = a_{0z} \cdot \alpha - \Delta C_z(\alpha) \tag{2.7}$$

where a_{0z} is the slope of the linear aerodynamic curve.

$-C_z$ can be approximated in any convenient way suitable for a particular study. For this model, a straight segment between discrete points will be used, as in Fig.2.2.

2.1.1. Harmonic decomposition of aerodynamic loads

Now, using the harmonic balance method, the system of differential equations (expressed in the time domain) is transformed into a system of non-linear algebraic equations (expressed in the frequency domain). Then it is necessary to write all the variables involved in the equations of the model as the sum of a mean, sine and cosine parts. For the pitching angle (θ) and deflection at the quarter chord (h):

$$\theta(\tau) = \theta_0 + \theta_s \cdot \sin k\tau + \theta_c \cdot \cos k\tau \tag{2.8}$$

$$h(\tau) = h_0 + h_s \cdot \sin k\tau + h_c \cdot \cos k\tau \tag{2.9}$$

Here ω represents the real frequency of vibration.

Their derivatives are:

$$\dot{\theta} = \theta_s \cdot k \cdot \cos k\tau - \theta_c \cdot k \cdot \sin k\tau \tag{2.10}$$

$$\ddot{\theta} = -\theta_s \cdot k^2 \cdot \sin k\tau - \theta_c \cdot k^2 \cdot \cos k\tau \tag{2.11}$$

$$\dot{h} = h_s \cdot k \cdot \cos k\tau - h_c \cdot k \cdot \sin k\tau \tag{2.12}$$

$$\ddot{h} = -h_s \cdot k^2 \cdot \sin k\tau - h_c \cdot k^2 \cdot \cos k\tau \tag{2.13}$$

The effective angle of attack is:

$$\alpha = \alpha_0 + \alpha_s \cdot \sin k\tau + \alpha_c \cdot \cos k\tau \tag{2.14}$$

or, in a purely sinusoidal form:

$$\alpha = \alpha_0 + \alpha_v \cdot \sin(k\tau + \xi) = \alpha_0 + \alpha_v \cdot \sin\phi \tag{2.15}$$

where
$$\alpha_v = \sqrt{\alpha_s^2 + \alpha_c^2} \tag{2.16}$$

$$\phi = k \cdot \tau + \xi \tag{2.17}$$

$$\xi = \sin^{-1}\frac{\alpha_c}{\alpha_v} \tag{2.18}$$

And as the incidence angle can be decomposed into pitching and plunging ($\alpha = \theta - \frac{\dot{\tilde{h}}}{}$):

$$\alpha_0 = \theta_0 \tag{2.19}$$

$$\alpha_s = \theta_s + \frac{h_c}{b} \cdot k \tag{2.20}$$

$$\alpha_c = \theta_c - \frac{h_s}{b} \cdot k \tag{2.22}$$

Following the same procedure for C_{zK} (circulatory contribution) and its first derivative results in:

$$C_{z\gamma} = C_{z\gamma 0} + C_{z\gamma s} \cdot \sin k\tau + C_{z\gamma c} \cdot \cos k\tau \tag{2.23}$$

$$\dot{C}_{z\gamma} = C_{z\gamma s} \cdot k \cdot \cos k\tau - C_{z\gamma c} \cdot k \cdot \sin k\tau \tag{2.24}$$

Substituting the equations obtained in that decomposition in eq. (2.3), it yields:

$$C_{z\gamma 0} = a_{0z} \cdot \theta_0 \tag{2.25}$$

$$C_{z\gamma s} = F(k) \cdot L_s - G(k) \cdot L_c \tag{2.26}$$

$$C_{z\gamma c} = G(k) \cdot L_s + F(k) \cdot L_c \tag{2.27}$$

where:

$$L_s = a_{0z} \cdot \left(\theta_s - k \cdot \frac{h_c}{b} - k \cdot \theta_c\right) \tag{2.28}$$

$$L_c = a_{0z} \cdot \left(\theta_c + k \cdot \frac{h_s}{b} + k \cdot \theta_s\right) \tag{2.29}$$

$$F(k) = \frac{k^2 \cdot \lambda_2 + \lambda_1^2}{\lambda_1^2 + k^2} \tag{2.30}$$

$$G(k) = \frac{k \cdot \lambda_1 \cdot (\lambda_2 - 1)}{\lambda_1^2 + k^2} \tag{2.31}$$

In the above, F and G are the real and imaginary parts, respectively, of C(k):

$C(k) = F(k) + i \cdot G(k)$

C(k) is the approximation to the Theodorsen function $C(ik) = \frac{\lambda_2 \cdot ik + \lambda_1}{ik + \lambda_1}$, which is characteristic of circulation terms, where λ_1=0.15, λ_2=0.55 and $i = \sqrt{-1}$.

Using the harmonic decomposition shown in eq. (2.25), (2.26) and (2.27) the following decomposition can be obtained for the linear part of the aerodynamic coefficient (C_{z1}):

Mean component:

$$C_{z10} = C_{z\gamma 0} \tag{2.32}$$

Sinus component:

$$C_{z1s} = S_{z1} \cdot \left(k \cdot \theta_c - \frac{h_s}{b} \cdot k^2 \right) - S_{z2} \cdot \theta_s \cdot k^2 - S_{z3} \cdot k \cdot \theta_c + C_{z\gamma s} \tag{2.33}$$

Cosinus component:

$$C_{z1c} = S_{z1} \cdot \left(k \cdot \theta_s - \frac{h_c}{b} \cdot k^2 \right) - S_{z2} \cdot \theta_c \cdot k^2 + S_{z3} \cdot k \cdot \theta_s + C_{z\gamma c} \tag{2.34}$$

2.1.2. Harmonic decomposition of the non-linear part

For the non-linear portion of the ONERA model, the same harmonic decomposition is used.

Since the aerodynamic curve is modelled by a single break point approximation, $-C_z$ can be calculated as

$$\begin{aligned} \Delta C_z &= b_1 \cdot (\alpha - \alpha_1) \quad \text{for } \alpha > \alpha_1 \\ &= 0 \qquad\qquad\quad \text{for } \alpha \leq \alpha_1 \end{aligned} \tag{2.35}$$

where I_1 is the stall angle and b_1 is the slope of the deviation from linear force curve.

To simplify the rest of the calculation the angle of attack (α) is used in the form where it is purely sinusoidal, then:

$$\Delta C_z = b_1 \cdot (\alpha_0 + \alpha_v \cdot \sin\phi - \alpha_1) \tag{2.36}$$

Using the decomposition in mean, sine and cosine terms:

$$\Delta C_z = \Delta C_{z0} + \Delta C_{zs1} \cdot \sin\phi + \Delta C_{zc2} \cdot \cos 2\phi \tag{2.37}$$

Only one sine term is included above due to the single-valued α function assumed for ΔC_z.

Substituting in the above using (2.36) and Fourier transformation gives:

$$\Delta C_{z0} = \frac{1}{\pi} \cdot \int_{\phi_1}^{\pi/2} b_1 \cdot (\alpha_0 + \alpha_v \cdot \sin\phi - \alpha_1)\, d\phi \tag{2.38}$$

$$\Delta C_{zs1} = \frac{2}{\pi} \cdot \int_{\phi_1}^{\pi/2} b_1 \cdot (\alpha_0 + \alpha_v \cdot \sin\phi - \alpha_1) \cdot \sin\phi\, d\phi \tag{2.39}$$

$$\Delta C_{zs2} = \frac{2}{\pi} \cdot \int_{\phi_1}^{\pi/2} b_1 \cdot (\alpha_0 + \alpha_v \cdot \sin\phi - \alpha_1) \cdot \cos 2\phi\, d\phi \tag{2.40}$$

which after appropriate manipulation results:

$$\Delta C_{z0} = \frac{b_1 \cdot \alpha_v}{\pi} \cdot \left[\left(\frac{\alpha_1 - \alpha_0}{\alpha_v} \right) \cdot \left(\phi_1 - \frac{\pi}{2} \right) + \cos\phi_1 \right] \tag{2.41}$$

$$\Delta C_{zs1} = \frac{b_1 \cdot \alpha_v}{\pi} \cdot \left[-\left(\phi_1 - \frac{\pi}{2} \right) - \frac{1}{2} \cdot \sin 2\phi_1 \right] \tag{2.42}$$

$$\Delta C_{zc2} = \frac{b_1 \cdot \alpha_v}{\pi} \cdot \left[-\frac{1}{2} \cdot \cos\phi_1 - \frac{1}{6} \cdot \sin 3\phi_1 \right] \tag{2.43}$$

where ϕ_1 is the non-dimensional phase associated with the angle of attack.

The value of ϕ_1 is calculated as follows:

$$\phi_1 = \sin^{-1}\delta \quad , -1<\delta<1 \text{ (partial stall)} \tag{2.41}$$

$$\phi_1 = \frac{\pi}{2} \quad , \delta>1 \text{ (no stall)} \tag{2.42}$$

$$\phi_1 = -\frac{\pi}{2} \quad , \delta<-1 \text{ (full stall)} \tag{2.43}$$

where $$\delta = \frac{\alpha_1 - \alpha_0}{\alpha_v} \tag{2.44}$$

If due to the oscillation amplitude, negative values of α are reached, the symmetric portion of the aerodynamic curve has to be considered. This is discussed in Appendix 1.

To obtain the final form of the non-linear part of the airloads, the expressions for ΔC_z and C_z are now included in eq.(2.4). After the substitution the RHS of eq.(2.4) is:

$$RHS = R_0 + R_{s1}\sin\phi + R_{c1}\cos\phi + R_{s2}\sin 2\phi + R_{c2}\cos 2\phi \tag{2.48}$$

where

$$R_0 = -r_2\Delta C_{z0} \tag{2.49}$$

$$R_{s1} = -r_2\Delta C_{zs1} \tag{2.50}$$

$$R_{c1} = -r_3 k\Delta C_{zs1} \tag{2.51}$$

$$R_{s2} = 2r_3 k\Delta C_{zc2} \tag{2.52}$$

$$R_{c2} = -r_2\Delta C_{zc2} \tag{2.53}$$

Matching terms of both sides of (2.4) gives

$$C_{z2} = B_{z0} + B_{zs1}\sin\phi + B_{zc1}\cos\phi + B_{zs2}\sin 2\phi + B_{zc2}\cos 2\phi \tag{2.54}$$

where:

$$B_{z0} = \frac{R_0}{r_2} \tag{2.55}$$

$$B_{zs1} = \frac{k_1 R_{s1} + k_2 R_{c1}}{k_1^2 + k_2^2} \tag{2.56}$$

$$B_{zc1} = \frac{k_1 R_{c1} - k_2 R_{s1}}{k_1^2 + k_2^2} \tag{2.57}$$

$$B_{zs2} = \frac{k_3 R_{s2} + k_4 R_{c2}}{k_3^2 + k_4^2} \tag{2.58}$$

$$B_{zc2} = \frac{k_3 R_{c2} - k_4 R_{s2}}{k_3^2 + k_4^2} \tag{2.59}$$

$$k_1 = r_2 - k^2 \qquad k_2 = r_1 k$$

$$k_3 = r_2 - 4k^2 \qquad k_4 = 2r_1 k \tag{2.60}$$

r_1, r_2 and r_3 are the parameters, which appear in the non-linear equation of the ONERA model, and have to be given to the model for every case. In this work, those parameters will be calculated by fitting the ONERA model to experimental data and to CFD calculations. This will be explained later on.

Now using the dimensionless time (τ) instead of the phase (ϕ) in (2.54), the following expression is obtained:

$$\begin{aligned} C_{z2} &= B_{z0} + B_{zs1}\cos\xi\sin k\tau + B_{zs1}\sin\xi\cos k\tau + \\ &+ B_{zc1}\cos\xi\cos k\tau - B_{zc1}\sin\xi\sin k\tau + \\ &+ B_{zs2}\cos 2\xi\sin 2k\tau + B_{zs2}\sin 2\xi\cos 2k\tau + \\ &+ B_{zc2}\cos 2\xi\cos 2k\tau - B_{zc2}\sin 2\xi\sin 2k\tau \end{aligned} \tag{2.61}$$

since $\phi = k\tau + \xi$.

As shown before:

$$\sin\xi = \frac{\alpha_c}{\alpha_v}, \quad \cos\xi = \frac{\alpha_s}{\alpha_v} \tag{2.62}$$

and thus:

$$C_{z2} = C_{z20} + C_{z2s1}\sin k\tau + C_{z2c1}\cos k\tau + C_{z2s2}\sin 2k\tau + C_{z2c2}\cos 2k\tau \tag{2.63}$$

where:

$$C_{z20} = B_{z0} \tag{2.64}$$

$$C_{z2s1} = B_{zs1}\frac{\alpha_s}{\alpha_v} - B_{zc1}\frac{\alpha_c}{\alpha_v} \tag{2.65}$$

$$C_{z2c1} = B_{zs1}\frac{\alpha_c}{\alpha_v} + B_{zc1}\frac{\alpha_s}{\alpha_v} \tag{2.66}$$

$$C_{z2s2} = B_{zs2}\left(\frac{\alpha_s^2}{\alpha_v^2} - \frac{\alpha_c^2}{\alpha_v^2}\right) - B_{zc2}\,2\frac{\alpha_c\cdot\alpha_s}{\alpha_v^2} \tag{2.67}$$

$$C_{z2c2} = B_{zc2}\left(\frac{\alpha_s^2}{\alpha_v^2} - \frac{\alpha_c^2}{\alpha_v^2}\right) + B_{zs2}\,2\frac{\alpha_c\cdot\alpha_s}{\alpha_v^2} \tag{2.68}$$

Now, with eq. (2.32), (2.33) and (2.34) the linear part of the aerodynamic coefficients can be calculated, and with eq. (2.61) the non-linear part. The total aerodynamic coefficient is the sum of both parts.

2.2. Flutter calculation

The flutter calculation is based in a modal analysis on plate deflection using the Ritz formulation (Rayleigh-Ritz)[2.2].

Assuming only out-of-plane deflection and rotational displacements (α, β):

$$\omega = \sum_{i=1}^{n} \gamma_i(x,y) q_i \tag{2.69}$$

$$\alpha = \theta_R + \sum_{i=1}^{n} \frac{\partial(\gamma_i(x,y))}{\partial y} q_i \tag{2.70}$$

where $\gamma_i(x,y)$ is the non-dimensional mode shape, q_i is the modal amplitude and n is the number of modes shapes.

Simplify $K_i(x,y)$ assuming:

$\gamma_i(x,y) = \phi_{ih}(x)$ bending

$\gamma_i(x,y) = \frac{y}{c}\phi_{ia}(x)$ torsion

The out-of-plane bending mode is written:

$$\phi_h(x) = \cosh\left(\varepsilon_1 \frac{x}{l}\right) - \cos\left(\varepsilon_1 \frac{x}{l}\right) - \alpha_1\left[\sinh\left(\varepsilon_1 \frac{x}{l}\right) - \sin\left(\varepsilon_1 \frac{x}{l}\right)\right] \tag{2.71}$$

where $\varepsilon_1 = \rho\pi = 1.87510$

$\rho = 0.5960$

$$\alpha_1 = \frac{\sinh\varepsilon_1 - \sin\varepsilon_1}{\cosh\varepsilon_1 + \cos\varepsilon_1} = 0.72664$$

and l is the wing length.

The torsion mode is:

$$\phi_a(x) = \sin\left(\frac{\pi x}{2l}\right) \tag{2.72}$$

The Rayleigh-Ritz potential method is now used to find the component of the mass and stiffness matrices.

$$T = \frac{1}{2}\iint m\dot{\omega}^2 dxdy = \frac{1}{2}\sum_i\sum_j \dot{q}_i\dot{q}_j M_{ij} \quad \text{(Kinetic energy)} \tag{2.73}$$

$$M_{ij} = \iint m\gamma_i\gamma_j dxdy \tag{2.74}$$

$$U = \frac{1}{2}\sum_i\sum_j q_i q_j k_{ij} \quad \text{(Internal energy)} \tag{2.75}$$

The expressions of T and U are substituted in Lagrange's equations of motion to give:

$$[M]\cdot\{\ddot{q}\} + [K]\cdot\{q\} = \{Q\} \tag{2.76}$$

{Q}can be derived from the work expression:

$$\delta W = \iint \delta w \Delta \rho dx dy = \sum_i \delta q_i Q_i \tag{2.77}$$

$$Q_i = \iint \gamma_i \Delta \rho dx dy \tag{2.78}$$

The components of the mass matrix [M] are:

$$M_{11} = MlI_1 \tag{2.79}$$

$$M_{22} = Ml\left(\frac{r_0}{c}\right)^2 I_3 \tag{2.80}$$

$$M_{12} = M_{21} = 0 \tag{2.81}$$

where:

$$I_1 = \int \phi_h^2 \, dx \tag{2.82}$$

$$I_3 = \int \phi_a^2 \, dx \tag{2.83}$$

and M is the mass per unit length and r_0 is the gyration radius.

The components of the stiffness matrix [K] are:

$$K_{11} = D_{11} \frac{c}{l^2} I_4 \tag{2.84}$$

$$K_{22} = D_{22} \frac{4}{lc} I_5 \tag{2.85}$$

where D_{11} and D_{22} are the flexural rigidity modulus and:

$$I_4 = \int \phi_h \, dx \tag{2.86}$$

$$I_5 = \int \phi_a \, dx \tag{2.87}$$

Then, the modal forces are:

$$Q_1 = \int_0^1 \phi_h L_{1/2} \, dx \tag{2.88}$$

$$Q_2 = \frac{1}{c} \int_0^1 \phi_a M_{a\,1/2} \, dx \tag{2.89}$$

where $L_{1/2}$ is the lift at the mid-chord and $M_{a1/2}$ is the moment at the mid-chord.

Using the nomenclature (2.6) in (2.76), it gives:

$$\mu\pi I_1\left(\ddot{\tilde{q}}_1 + \Theta^2 k_a^2 \tilde{q}_1\right) = \int_0^1 \phi_h C_{L1/2}\, d\tilde{x} \tag{2.90}$$

$$\mu\frac{\pi}{4} r_a^2 I_3\left(\ddot{\tilde{q}}_2 + k_a^2 \tilde{q}_2\right) = \int_0^1 \phi_a C_{M1/2}\, d\tilde{x} \tag{2.91}$$

where:

$$\Theta = \frac{\omega_h}{\omega_a} \text{ is the ratio of bending to torsional frequency} \tag{2.92}$$

$$k_a = \frac{\omega_a b}{U} \text{ is the torsional reduced frequency} \tag{2.93}$$

$$\mu = \frac{M}{\pi\rho b^2} \text{ is the density ratio} \tag{2.94}$$

$$r_a = \frac{r_0}{b} = \frac{\sqrt{I_a / M}}{b} \text{ , being } I_a \text{ the moment of inertia.} \tag{2.95}$$

All aero-forces are calculated at the quarter-chord but the structural components are by convention formulated with the longitudinal wing axes placed at the mid-chord. Thus the aero-forces and the coordinates as well are to be converted (Fig.2.4).

$$L_{1/2}=L_{1/4} \tag{2.96}$$

$$M_{a1/2} = M_{a1/4} + \frac{b}{2} L_{1/4} \tag{2.97}$$

$$\theta_{1/2} = \theta_R + \frac{1}{2}\phi_a \tilde{q}_2 \tag{2.98}$$

$$h_{1/2} = \phi_h b \tilde{q}_1 \tag{2.99}$$

$$h_{1/4} = h_{1/2} + \frac{b}{2}\theta_{1/2} \tag{2.100}$$

where θ_R is the root angle.

Using (2.98) and (2.99) in (2.100) and noting that the pitch angle (θ) remains the same, the following expressions are obtained for the tip deflection (h) and pitch angle at the quarter-chord:

$$h_{1/4} = b\left(\frac{1}{2}\theta_R + \phi_h \tilde{q}_1 + \frac{1}{4}\phi_a \tilde{q}_2\right) \tag{2.101}$$

$$\theta_{1/4} = \theta_{1/2} = \theta_R + \frac{1}{2}\phi_a \tilde{q}_2 \tag{2.102}$$

Now, harmonic variation in time is assumed for $\tilde{q}_1$ and $\tilde{q}_2$.

$$\tilde{q}_1 = \tilde{q}_{10} + \tilde{q}_{1s}\sin k\tau + \tilde{q}_{1c}\cos k\tau \tag{2.103}$$

$$\tilde{q}_2 = \tilde{q}_{20} + \tilde{q}_{2s}\sin k\tau + \tilde{q}_{2c}\cos k\tau \tag{2.104}$$

And using those formulations in (2.101) and (2.102), it remains:

$$h_{1/4} = h_0 + h_s \sin k\tau + h_c \cos k\tau \tag{2.105}$$

where

$$h_0 = b\left(\frac{\theta_R}{2} + \phi_h \tilde{q}_{10} + \frac{1}{4}\phi_a \tilde{q}_{20}\right) \tag{2.106}$$

$$h_s = b\left(\phi_h \tilde{q}_{1s} + \frac{1}{4}\phi_a \tilde{q}_{2s}\right) \tag{2.107}$$

$$h_c = b\left(\phi_h \tilde{q}_{1c} + \frac{1}{4}\phi_a \tilde{q}_{2c}\right) \tag{2.108}$$

and

$$\theta_{1/4} = \theta_0 + \theta_s \sin k\tau + \theta_c \cos k\tau \tag{2.109}$$

where

$$\theta_0 = \theta_R + \frac{1}{2}\phi_a \tilde{q}_{20} \tag{2.110}$$

$$\theta_s = \frac{1}{2}\phi_a \tilde{q}_{2s} \tag{2.111}$$

$$\theta_c = \frac{1}{2}\phi_a \tilde{q}_{2c} \tag{2.112}$$

Substituting in the expression for the linear coefficient (2.32, 2.33, 2.34) the results of the above decomposition, the aerodynamic coefficient (C_{z1}) can be expressed as a function of the harmonic terms of $\tilde{q}_1$ and $\tilde{q}_2$:

$$C_{z10} = a_{0z}\left(\theta_R + \frac{1}{2}\phi_a \tilde{q}_{20}\right) \tag{2.113}$$

$$C_{z1s} = \phi_h\left(E_{z1}\tilde{q}_{1s} + E_{z2}\tilde{q}_{1c}\right) + \phi_a\left(E_{z3}\tilde{q}_{2s} + E_{z4}\tilde{q}_{2c}\right) \tag{2.114}$$

$$C_{z1c} = \phi_h\left(-E_{z2}\tilde{q}_{1s} + E_{z1}\tilde{q}_{1c}\right) + \phi_a\left(-E_{z4}\tilde{q}_{2s} + E_{z3}\tilde{q}_{2c}\right) \tag{2.115}$$

where:

$$E_{z1} = S_{z1}k^2 + a_{0z}G(k)k \tag{2.116}$$

$$E_{z2} = a_{0z}F(k)k \tag{2.117}$$

$$E_{z3} = \frac{1}{4}S_{z1}k^2 - \frac{1}{2}S_{z2}k^2 + \frac{1}{2}a_{0z}F(k) - \frac{1}{4}a_{0z}kG(k) \tag{2.118}$$

$$E_{z4} = -\frac{1}{2}S_{z1}k - \frac{1}{2}S_{z3}k - \frac{1}{4}a_{0z}kF(k) - \frac{1}{2}a_{0z}G(k) \tag{2.119}$$

To calculate the non-linear part (C_{z2}) the angle of incidence (α) is transformed in structural coordinates:

$$\alpha_{1/4} = \alpha_0 + \alpha_s \sin(k\tau) + \alpha_c \cos(k\tau) \tag{2.120}$$

$$\alpha_0 = \theta_R + \frac{1}{2}\phi_a \tilde{q}_{20} \tag{2.121}$$

$$\alpha_s = \frac{1}{2}\phi_a \tilde{q}_{2s} + \phi_h k\tilde{q}_{1c} + \frac{1}{4}\phi_a k\tilde{q}_{2c} \tag{2.122}$$

$$\alpha_c = \frac{1}{2}\phi_a \tilde{q}_{2c} - \phi_h k\tilde{q}_{1s} - \frac{1}{4}\phi_a k\tilde{q}_{2s} \tag{2.122}$$

and the above is to be used in the eq. (2.61) to have the final expression for the non-linear part (C_{z2}).

2.3. Combined structural-aerodynamics equations - Flutter equations.

Using the harmonic balance method, the forcing terms of the flutter equations (2.90 ,2.91) can be written as function of their harmonics.

The transformed force components are inserted in the forcing terms (2.90,2.91) using (2.96,2.97).

$$\int_0^1 \phi_h C_{L1/2} d\tilde{x} = \int_0^1 \phi_h \left(C_{L10} + C_{L1s1}\sin k\tau + C_{L1c1}\cos k\tau\right) d\tilde{x} + \int_0^1 \phi_h \left(C_{L20} + C_{L2s1}\sin k\tau + C_{L2c1}\cos k\tau\right) d\tilde{x} \tag{2.124}$$

$$\int_0^1 \phi_a C_{M1/2}\,d\tilde{x} = \int_0^1 \phi_a \left(\left(C_{M10} + \frac{1}{4} C_{L10} \right) + \left(C_{M1s1} + \frac{1}{4} C_{L1s1} \right) \sin k\tau + \right.$$
$$\left. + \left(C_{M1c1} + \frac{1}{4} C_{L1c1} \right) \cos k\tau \right) d\tilde{x} +$$
$$+ \int_0^1 \phi_a \left(\left(C_{M20} + \frac{1}{4} C_{M20} \right) + \left(C_{M2s1} + \frac{1}{4} C_{L2s1} \right) \sin k\tau + \right. \qquad (2.125)$$
$$\left. + \left(C_{M2c1} + \frac{1}{4} C_{L2c1} \right) \cos k\tau \right) d\tilde{x}$$

Now, using (2.113, 2.114, 2.115) and (2.61) in the last two equations:

$$\int_0^1 \phi_h C_{L1/2}\,d\tilde{x} = I_4 a_{0L} \theta_R + I_2 \frac{a_{0L}}{2} \tilde{q}_{20} +$$
$$+ \left(I_1 \left(E_{L1} \tilde{q}_{1s} + E_{L2} \tilde{q}_{2s} \right) + I_2 \left(E_{L3} \tilde{q}_{2s} + E_{L4} \tilde{q}_{2c} \right) \right) \sin k\tau +$$
$$+ \left(I_1 \left(-E_{L2} \tilde{q}_{2s} + E_{L1} \tilde{q}_{1c} \right) + I_2 \left(-E_{L4} \tilde{q}_{2s} + E_{L3} \tilde{q}_{2c} \right) \right) \cos k\tau + \qquad (2.126)$$
$$+ I_4 B_{L0} + \left(I_4 \left(B_{Ls1} \frac{\alpha_s}{\alpha_v} - B_{Lc1} \frac{\alpha_c}{\alpha_v} \right) \right) \sin k\tau +$$
$$+ \left(I_4 \left(B_{Ls1} \frac{\alpha_c}{\alpha_v} + B_{Lc1} \frac{\alpha_s}{\alpha_v} \right) \right) \cos k\tau$$

$$\int_0^1 \phi_a C_{M1/2}\,d\tilde{x} = I_5 a_{0T} \theta_R + I_3 \frac{a_{0T}}{2} \cdot \tilde{q}_{20} +$$
$$+ \left(I_2 \left(E_{T1} \tilde{q}_{1s} + E_{T2} \tilde{q}_{2c} \right) + I_3 \left(E_{T3} \tilde{q}_{2s} + E_{T4} \tilde{q}_{2c} \right) \right) \sin k\tau +$$
$$+ \left(I_2 \left(-E_{T2} \tilde{q}_{1s} + E_{T1} \tilde{q}_{1c} \right) + I_3 \left(-E_{T4} \tilde{q}_{2s} + E_{T3} \tilde{q}_{2c} \right) \right) \cos k\tau + \qquad (2.127)$$
$$+ I_5 B_{T0} + \left(I_5 \left(B_{Ts1} \frac{\alpha_s}{\alpha_v} - B_{Tc1} \frac{\alpha_c}{\alpha_v} \right) \right) \sin k\tau +$$
$$+ \left(I_5 \left(B_{Ts1} \frac{\alpha_c}{\alpha_v} + B_{Tc1} \frac{\alpha_s}{\alpha_v} \right) \right) \cos k\tau$$

where:

$$I_2 = \int_0^1 \phi_h \phi_a\,d\tilde{x} \qquad (2.128)$$

and I_1, I_3, I_4 and I_5 are already defined in eq. (2.82,2.83,2.86,2.87).

L subscript denotes components related to the lift coefficient and I_x are the aerodynamic integrals.

T subscript denotes components related to transformed values of the moment coefficients.

$$\alpha_{0T} = \alpha_{0M} + \frac{\alpha_{0L}}{4} \qquad E_{T1} = E_{M1} + \frac{E_{L1}}{4} \qquad E_{T2} = E_{M2} + \frac{E_{L2}}{4}$$

$$E_{T3} = E_{M3} + \frac{E_{L3}}{4} \qquad E_{T4} = E_{M4} + \frac{E_{L4}}{4} \qquad B_{T0} = B_{M0} + \frac{B_{L0}}{4}$$

$$B_{Ts1} = B_{Ms1} + \frac{B_{Ls1}}{4} \qquad B_{Tc1} = B_{Mc1} + \frac{B_{Lc1}}{4} \tag{2.129}$$

Now using the harmonic decomposition of $\tilde{q}_1$ and $\tilde{q}_2$ in the LHS of (2.90, 2.91) and (2.126, 2.127) in the LHS, matching terms of both sides gives 6 equations:

$$\mu\pi I_1 \Theta^2 K_a^2 \tilde{q}_{10} = I_4 a_{0L} \theta_R + I_2 \frac{a_{0L}}{2} \tilde{q}_{20} + I_4 B_{L0} \tag{2.130}$$

$$\mu \frac{\pi}{4} r_a^2 I_3 K_a^2 \tilde{q}_{20} = I_5 a_{0T} \theta_R + I_3 \frac{a_{0T}}{2} \cdot \tilde{q}_{20} + I_5 B_{T0} \tag{2.131}$$

$$\begin{aligned} \mu\pi I_1 (\Theta^2 K_a^2 - k^2) \tilde{q}_{1s} = {} & I_1 \left(E_{L1} \tilde{q}_{1s} + E_{L2} \tilde{q}_{1c} \right) + I_2 \left(E_{L3} \tilde{q}_{2s} + E_{L4} \tilde{q}_{2c} \right) + \\ & + I_4 \left(B_{Ls1} \frac{\alpha_s}{\alpha_v} - B_{Lc1} \frac{\alpha_c}{\alpha_v} \right) \end{aligned} \tag{2.132}$$

$$\begin{aligned} \mu\pi I_1 (\Theta^2 K_a^2 - k^2) \tilde{q}_{1c} = {} & I_1 \left(-E_{L2} \tilde{q}_{1s} + E_{L1} \tilde{q}_{1c} \right) + I_2 \left(-E_{L4} \tilde{q}_{2s} + E_{L3} \tilde{q}_{2c} \right) + \\ & + I_4 \left(B_{Lc1} \frac{\alpha_s}{\alpha_v} + B_{Ls1} \frac{\alpha_c}{\alpha_v} \right) \end{aligned} \tag{2.133}$$

$$\begin{aligned} \mu \frac{\pi}{4} r_a^2 I_3 (K_a^2 - k^2) \tilde{q}_{2s} = {} & I_2 \left(E_{r1} \tilde{q}_{1s} + E_{r2} \tilde{q}_{1c} \right) + I_3 \left(E_{r3} \tilde{q}_{2s} + E_{r4} \tilde{q}_{2c} \right) + \\ & + I_5 \left(B_{rs1} \frac{\alpha_s}{\alpha_v} - B_{rc1} \frac{\alpha_c}{\alpha_v} \right) \end{aligned} \tag{2.134}$$

$$\begin{aligned} \mu \frac{\pi}{4} r_a^2 I_3 (K_a^2 - k^2) \tilde{q}_{2c} = {} & I_2 \left(-E_{r2} \tilde{q}_{1s} + E_{r1} \tilde{q}_{1c} \right) + I_3 \left(-E_{r4} \tilde{q}_{2s} + E_{r3} \tilde{q}_{2c} \right) + \\ & + I_5 \left(B_{rc1} \frac{\alpha_s}{\alpha_v} + B_{rs1} \frac{\alpha_c}{\alpha_v} \right) \end{aligned} \tag{2.135}$$

This set of equations has to be solved using the Newton-Raphson method, as is explained in Appendix 2. After that, the flutter velocity (V_F) and frequency (f_F) are calculated:

$$V_F = \frac{\omega_a \cdot b}{k_a} \tag{2.136}$$

$$f_F = \frac{V_F}{2\pi} \frac{k}{b} \tag{2.137}$$

Chapter 3. Results

A NACA-0012 aerofoil was used in this study. Detailed experimental data in the form of aerodynamic coefficients (C_L, C_M, C_D) and steady and unsteady surface pressure coefficients (C_p) are available in [31].

Two different cases were. The first one corresponds to the aerofoil executing a harmonic motion of 0° of mean angle and 5° of amplitude. This case is set up to correspond with the experimental data available in [31]. The second case corresponds to a motion of 9.97° of mean angle and 9.88° of amplitude. The reduced frequency is k=0.145, the Mach number is M=0.302 and the Reynolds number is Re=3.67e6, for all the cases. All these values are summarised in Table 3.1. Experimental data are only available for the first case.

A 150 x 50 point grid was used for the computation. The first grid point away from the surface was located at a distance of $10^{-6} \cdot c$, where c is the chord length.

The results were calculated by first computing a steady flow solution, used as initial guess in the unsteady fluid solver. Then about three oscillatory cycles were computed in the unsteady solver. The CPU time was about 9 hours per cycle on a SGI workstation.

The first case was solved three times, using the k-ω turbulence model. Once the fluid solver calculations were done, the results were used to estimate the parameters r_1, r_2 and r_3 of the ONERA model. This was made by fitting the results from the ONERA model to the results from the fluid solver, using the Levenberg-Marquardt method [3.2, 3.3] (see Appendix 3), a non-linear fitting method. The fitting is used for the experimental data corresponding to the first case as well. The parameters are assumed to be in the form:

$$r_1 = r_{10} + r_{12}\Delta C_{z0}^2$$

$$r_2 = \left[r_{20} + r_{22}\Delta C_{z0}^2\right]^2$$

$$r_3 = \left[r_{30} + r_{32}\Delta C_{z0}^2\right] r_2$$

Then, six parameters are to be fitted for each case: r_{10}, r_{12}, r_{20}, r_{22}, r_{30} and r_{32}. The fitting was only made for the lift coefficient. But due to the calculation of the transformed forces from 1/4 chord to 1/2 chord, the r_i coefficients obtained for the lift can be used for the moment as well. The parameters obtained with the fitting are summarised in Table 3.2. A harmonic decomposition of the lift experimental data, the lift fluid solver loops and the fitting results was calculated in order to compare them. The values of the decomposition are shown in Table 3.3. It can be seen that fair agreement is obtained. Further comparison is done to show the agreement between the computational lift loops and the fitted values predicted by the ONERA model. These comparisons can be seen in Figs. 3.1-3.2. for the corresponding cases 1 and 2. Included in these figures is the CFD coefficient of lift versus angle of attack plus the linear and non linear parts of the ONERA model. The ONERA model lift curve slope shows good agreement with the CFD predictions for case 1. For case 2, the lift curve slope is on the up-stoke of the motion is well captured, and the stall angle shows good agreement. The lift recovery is good, but the ONERA model cannot predict the variations in the lift due to vortex shedding that can be observed in the CFD predictions.

Once the fitting was done, the flutter program is run for all the cases using the parameters r_{10}, r_{12}, r_{20}, r_{22}, r_{30} and r_{32} estimated before. Other constants and parameters used in the flutter calculations are described in Table 3.4. The oscillation frequency versus the flutter frequency has been plotted in Fig. 3.3. This shows the rapid increase in the oscillations at the onset of flutter as is expected.

Chapter 4. Conclusions and suggestions for future work

4.1. Conclusions

A simple analytical method to include non-linear aerodynamic effects (by means of the ONERA model) into a stall flutter analysis was developed. The analysis applies the mathematical tools of Fourier analysis, harmonic balance and the Newton-Raphson as a numerical solver. The PMB fluid dynamics solver was used to calculate the airloads (lift) over an oscillating aerofoil using the k-ω turbulence model. The solutions compared well with experimental results but further work must be carried to assess the impact of the turbulence model used on the accuracy of the computational results. A numerical method was used to fit the ONERA model to the fluid solver results. The fitting was satisfactory, but due to the simplicity of the aeroelastic model the curves obtained were very simple, just ellipses. The flutter calculations were qualitative satisfactory as well, predicting the velocity and frequency of the onset of flutter.

4.2. Suggestions for future work

The computational grid used in this study was not very fine (150x50), and some improvements over the present calculations would be done if working with a finer grid. The time resolution used in the fluid solver was the minimum possible, and in future works higher values might be used.

For stall cases, it might be good to fit the ONERA model for the moment coefficient instead of doing it for the lift, as was done here. And, definitely, the quality of the fitting would be improved by increasing the number of parameters.

There is scope to use this method of predicting aerodynamic loads within flight dynamics codes as a replacement to look-up tables, which are currently used to give aerofoil loadings. This model can be extended to predict three-dimensional aerodynamics on finite wings and rotor-blades, and hence improve prediction methods for flutter analysis.

To further validate the code in the form it is currently, a set of experimental data should be obtained, including both aerodynamic loads and flutter characteristics, and a comparison between the prediction and the experiments carried out.

References

1.1. Fung, Y.C., "An Introduction to the Theory of Aeroelasticity", Dover Publications, 1993.

1.2. Tran, C.T. and Petot, D., "Semi-Empirical Model for the Dynamic Stall of Aerofoils in View of Application to the Calculation of Responses of a Helicopter in Forward Flight", *Vertica*, Vol.5, No.1, 1981, pp. 35-53.

1.3. Dat, D. and Tran, C.T., "Investigation of the Stall Flutter of an Aerofoil with a Semi-Empirical Model of 2-D Flow", *Vertica*, Vol.7, No.2., 1983, pp. 73-86.

1.4. Marshall, J.G. and Imregun, M., "A Review of Aeroelasticity Methods with Emphasis on Turbomachinery Applications", *Journal of Fluids and Structures*, 10, 1996, pp. 237-267.

1.5. Peters, D.A., "Toward a Unified Lift Model for Use in Rotor Blade Stability Analyses", *Journal of the American Helicopter Society*, Vol.30, No.3, 1985, pp. 32-42.

1.6. Petot, D. and Dat, R., "Unsteady Aerodynamic Loads on an Oscillating Aerofoil with Unsteady Stall", *Proceedings of 2nd Workshop on Dynamics and Aeroelasticity Stability Modelling of Rotorcraft Systems*, Florida Atlantic Univ., Boca Raton, FL, Nov. 1987.

1.7. Dunn, P. and Dugundji, J., "Nonlinear Stall Flutter and Divergence Analysis of Cantilevered Graphite-Epoxy Wings", *AIAA Journal*, Vol.30, No.1, 1992, pp. 153-162.

1.8. Chopra, I. and Dugundji, J., "Non-linear Dynamic Response of a Wind Turbine Blade", *Journal of Sound and Vibration*, Vol.63, No.2, 1979, pp. 265-286.

1.9. Tang, D. and Dowell, E., "Experimental and Theoretical Study for Non-linear Aeroelastic Behaviour of Flexible Rotor Blades", *AIAA Paper 92-2253*, April 1992.

1.10 Hodges, D. and Dowell, E., "Non-linear Equations of Motion for the Elastic Bending and Torsion of Twisted Non-uniform Rotor Blades", *NASA TN D-7818*, Dec. 1974.

1.11. Kim, T. and Dugundji, J., "Non-linear Large Amplitude Aeroelastic Behaviour of Composite Rotor Blades", *AIAA Journal*, Vol.31, No.8, 1993, pp. 1489-1497.

1.12. Abdel-Rahim, A., Sisto, F. and Thangam, S., "Computational Study of Stall Flutter in Linear Cascades", *ASME Journal of Turbomachinery*, Vol. 115, 1993, pp. 157-166.

1.13. Jang, H.M., Ekaterinaris, J.A., Platzer, M.F. and Cebeci, T., "Essential Ingredients for the Computation of Steady and Unsteady Boundary Layers", *ASME Journal of Turbomachinery*, Vol.113, 1991, pp.608-616.

1.14. Cebeci, T., Platzer, M.F., Jang, H.M., and Chen, H.H., "An Inviscid-Viscous Interaction Approach to the Calculation of Dynamic Stall Initiation on Aerofoils", *ASME Journal of Turbomachinery*, Vol.115, 1993, pp. 714-723.

1.15. Clarkson, J.D., Ekaterinaris, J.A., and Platzer, M.F., "Computational Investigation of Aerofoil Stall Flutter", *Unsteady Aerodynamics, Aeroacoustics and Aeroelasticity of Turbomachines and Propellers*, H.M. Atassi, ed. Springer-Verlag, New York, 1993, pp. 415-432.

1.16. McCroskey, W.J., "The Phenomenon of Dynamic Stall", *NASA TM-81264*, Mar. 1981.

1.17. Srinivasan, G.R., Ekaterinaris, J.A. and McCroskey, W.J., "Dynamic Stall of an Oscillating Wing, Part 1: Evaluation of Turbulence Models", *AIAA Paper 93-3403*, August 1993.

1.18. Ekaterinaris, J.A. and Menter, F.R., "Computation of Separated and Unsteady Flows with One and Two Equation Turbulence Models", *AIAA Paper 94-0190*, January 1994.

1.19. Ekaterinaris, J.A., Chandrasekhara, M.S. and Platzer, M.F., "Analysis of Low Reynolds Number Aerofoils", *AIAA Paper 94-0534*, January 1994.

1.20. Ekaterinaris, J.A. and Platzer, M.F., "Numerical Investigation of Stall Flutter", *Journal of Turbomachinery*, Vol. 118, April 1996, pp. 197-203.

1.21. Carta, F.O. and Lorber, P.F., "Experimental Study of the Aerodynamics of Incipient Tortional Stall Flutter", *Journal of Propulsion*, Vol.3., No.2, 1987, pp. 164-170.

1.22. McCroskey, W.J., McAlister, K.W, Carr, L.W. and Pucci, S.L., "An Experimental Study of Dynamic Stall on Advanced Aerofoil Sections. Volume 1: Summary of the Experiment", *NASA-TM-84245-VOL-1*, 1982.

1.23. Carr, L.W., "Progress in Analysis and Prediction of Dynamic Stall", *J. Aircraft*, 25 (1), Jan. 1988 pp. 6-17.

1.24. Visbal, M.R., "Effect of Compressibility on Dynamic Stall", *AIAA Paper 88-0132*, 1988.

1.25. Piziali, R.A., "An Experimental Investigation of 2D and 3D Oscillating Wing Aerodynamics for a range of angle of Attack including Stall", *NASA-TM-4632* (1993).

1.26. Ekaterinaris, J.A., Cricelli, A.S. and Platzer, M.F., "A Zonal Method for Unsteady Viscous Compressible Aerofoil Flows", *Journal of Fluids and Structures*, 8, pp. 107-123 (1994).

1.27. Mehta, U.B. and Lavan, Z., "Starting Vortex, Separation Bubbles and Stall: A Numerical Study of Laminar Unsteady Flow Around an Aerofoil", *J. Fluid Mech.*, 67(2), pp. 227-256 (1975).

1.28. Gulcat, U., "Separate Numerical Treatment of Attached and Detached Flow Regions in General Viscous Flows", Ph.D. Thesis, Georgia Institute of Technology, Atlanta, Georgia, 1981.

1.29. Wu, J.C., "Zonal Solution of Unsteady Viscous Flow Problems", *AIAA Paper 84-1637*, 1984.

1.30. Wu, J.C., Wang, C.M. and Tuncer, I.H., "Unsteady Aerodynamics of Rapidly Pitched Aerofoil", *AIAA Paper 86-1105*, 1986.

1.31. Tuncer, I.H., "Unsteady Aerodynamics of Oscillating and Rapidly Pitched Aerofoils", Ph.D. Thesis, Georgia Institute of Technology, Atlanta, Georgia, 1988.

2.1. Tran, C.T. and Petot, D., "Semi-Empirical Model for the Dynamic Stall of Aerofoils in View of Application to the Calculation of Responses of a Helicopter in Forward Flight," Vertica, Vol.5, No.1, 1981, pp. 35-53.

2.2. Meirovitch, L., "Elements of vibration analysis", Mc-Graw Hill International Editions, 2nd edition, 1986.

3.1. McAllister, K.W., Pucci, S.L., McCroskey, W.J. and Car, L.W., "An Experimental Study of Dynamic Stall on Advanced Aerofoil Sections, vol. 2: Pressure and Force Data", *NASA TM-84245*, September 1982.

3.2. Press, W.H., Teukolsky, S.A., Vetterling, W.T. and Flannery, B.P., "Numerical recipes in FORTRAN. The art of scientific computing", 2nd edition, Cambridge University Press, 1992.

3.3. Marquardt, D.W., *Journal of the Society for Industrial and Applied Mathematics*, vol.11, pp. 431-441, 1963.

Appendix 1

The symmetric part of the aerodynamic curves

If due to the oscillation amplitude negative values are reached, the symmetric portion of the aerodynamic curve has to be considered.

Using $\bar{\alpha}_1 = -\alpha_1$, the above three equations are expanded:

$$\Delta C_{z0} = \frac{b_1 \cdot \alpha_v}{\pi} \cdot \left[\left(\frac{\alpha_1 - \alpha_0}{\alpha_v} \right) \cdot \left(\phi_1 - \frac{\pi}{2} \right) + \cos \phi_1 \right] - \frac{b_1 \cdot \alpha_v}{\pi} \cdot \left[\left(\frac{\bar{\alpha}_1 - \alpha_0}{\alpha_v} \right) \cdot \left(\bar{\phi}_1 + \frac{\pi}{2} \right) + \cos \bar{\phi}_1 \right] \tag{A1.1}$$

$$\Delta C_{zs1} = \frac{b_1 \cdot \alpha_v}{\pi} \cdot \left[-\left(\phi_1 - \frac{\pi}{2} \right) - \frac{1}{2} \cdot \sin 2\phi_1 \right] - \frac{b_1 \cdot \alpha_v}{\pi} \cdot \left[-\left(\bar{\phi}_1 + \frac{\pi}{2} \right) - \frac{1}{2} \cdot \sin 2\bar{\phi}_1 \right] \tag{A1.2}$$

$$\Delta C_{zc2} = \frac{b_1 \cdot \alpha_v}{\pi} \cdot \left[-\frac{1}{2} \cdot \cos \phi_1 - \frac{1}{6} \cdot \sin 3\phi_1 \right] - \frac{b_1 \cdot \alpha_v}{\pi} \cdot \left[-\frac{1}{2} \cdot \cos \bar{\phi}_1 - \frac{1}{6} \cdot \sin 3\bar{\phi}_1 \right] \tag{A1.3}$$

Again:

$$\bar{\phi}_1 = \sin^{-1} \delta \quad , -1 < \bar{\delta} < 1$$

$$\bar{\phi}_1 = \frac{\pi}{2} \quad , \bar{\delta} > 1$$

$$\bar{\phi}_1 = -\frac{\pi}{2} \quad , \bar{\delta} < -1 \tag{A1.4}$$

where $\bar{\delta} = \dfrac{\bar{\alpha}_1 - \alpha_0}{\alpha_v}$

Appendix 2

The Newton-Raphson method

The system of six equations (2.130-2.135) is solved by means of the Newton-Raphson method.

First of all, the equations are written in the form:

$$F_i(x_1,...x_6)=0$$

The vector $(x_1,...x_6)$ is called state vector. Here, it should include the harmonic components of the modal amplitudes q_{i0}, q_{is} and q_{ic}. But, due to the possibility that the iterative solution procedure converges to a trivial solution (zero), a different approach is taken.

q_{2s} (sine component of second mode) is set to a small constant which gives the amplitude level of the oscillation, and q_{2c} (cosine component) is set to zero since for limit cycle oscillations we may consider any initial phase. The second mode (torsion) was chosen since this mode dominates the oscillation behaviour.

To complete the vector of the unknowns, the missing components are replaced by k_a (reduced torsional frequency) and k (reduced frequency).

The Newton-Raphson method is used:

$$F_i(x+\delta x) = F_i(x) + \sum_{j=1}^{6} \frac{\partial F_i}{\partial x_j} \cdot \delta x_j + H.O.T. = 0$$

The summation is the Jacobian matrix. It is calculated numerically, increasing the state vector by a small number. Then $J = \frac{\partial F}{\partial x_j} = \frac{\Delta F}{\Delta x_j}$.

Then, the correction is: $\delta x = -J^{-1} \cdot F(x)$, at the nth iteration. And the estimation of the vector at the next iteration is: $x_{n+1} = x_n + \delta x$.

For the present work 25 iterations are performed and if the change in variables is less than 10^{-4} the iteration procedure is considered as divergent.

Appendix 3

The Levenberg-Marquardt method

Fitting is considered when the model depends non-linearly on the set of M unknown parameters a_k, k=1, 2, ..., M. A merit function χ^2 is defined and the parameters are to be determined by its minimization. As there are non-linear dependences, the minimization must proceed iteratively. Given trial values for the parameters, a procedure is developed to improve the trial solution. The procedure is repeated until χ^2 stops decreasing.

Sufficiently close to the minimum, χ^2 can be approximated by a quadratic form:

$$\chi^2(\mathbf{a}) \approx \gamma - \mathbf{d}\cdot\mathbf{a} + \frac{1}{2}\mathbf{a}\cdot\mathbf{D}\cdot\mathbf{a} \qquad \text{(A3.1)}$$

where

$$\mathbf{d} = -\nabla\chi^2(\mathrm{a}) \text{ (an M-vector)} \qquad \text{(A3.2)}$$

$$\mathbf{D} = \frac{\partial^2\chi^2}{\partial a_k \partial a_l} \text{ (an MxM matrix, called Hessian matrix)} \qquad \text{(A3.3)}$$

If the approximation is a good one, one can jump from the current trial parameters $\mathbf{a}_{cur}$ to the minimizing ones $\mathbf{a}_{min}$ using:

$$\mathbf{a}_{min} = \mathbf{a}_{cur} + \mathbf{D}^{-1}\cdot\left[-\nabla\chi^2(\mathbf{a}_{cur})\right] \qquad \text{(A3.4)}$$

On the other hand, (A5.1) might be a poor local approximation to the shape of the function that is to be minimize at $\mathbf{a}_{cur}$. In that case, about all that can be done is take a step down the gradient, as in the steepest decent method [3.2]. In other words,

$$\mathbf{a}_{next} = \mathbf{a}_{cur} - \mathrm{constant}\times\nabla\chi^2(\mathbf{a}_{cur}) \qquad \text{(A3.5)}$$

where the constant has to be small enough not to exhaust the downhill direction.

To use (A5.4) or (A5.5) the gradient of the χ^2 function has to be computed at any set of parameters **a**. To use (A5.4) the matrix **D** is also needed, which is the second derivative matrix (Hessian matrix) of the χ^2 merit function, at any **a**.

If the model to be fitted is

$$y=y(x;\mathbf{a}) \tag{A3.6}$$

and the merit function is

$$\chi^2(\mathbf{a}) = \sum_{i=1}^{N}\left[\frac{y_i - y(x_i;\mathbf{a})}{\sigma_i}\right]^2 \tag{A3.7}$$

where σ_i is the standard deviation. Then the gradient of χ^2 has components:

$$\frac{\partial\chi^2}{\partial a_k} = -2\sum_{i=1}^{N}\frac{[y_i - y(x_i;\mathbf{a})]}{\sigma_i^2}\frac{\partial y(x_i;\mathbf{a})}{\partial a_k} \qquad k=1,2,...,M \tag{A3.8}$$

Taking an additional partial derivation gives:

$$\frac{\partial^2\chi^2}{\partial a_k \partial a_l} = 2\sum_{i=1}^{N}\frac{1}{\sigma_i^2}\left[\frac{\partial y(x_i;\mathbf{a})}{\partial a_k}\frac{\partial y(x_i;\mathbf{a})}{\partial a_l} - [y_i - y(x_i;\mathbf{a})]\frac{\partial^2 y(x_i;\mathbf{a})}{\partial a_l \partial a_k}\right] \tag{A3.9}$$

The second derivative term tends to cancel out when summed over i, then it remains:

$$\frac{\partial^2\chi^2}{\partial a_k \partial a_l} = 2\sum_{i=1}^{N}\frac{1}{\sigma_i^2}\left[\frac{\partial y(x_i;\mathbf{a})}{\partial a_k}\frac{\partial y(x_i;\mathbf{a})}{\partial a_l}\right] \tag{A3.10}$$

It is conventional to remove the factors of 2 by defining

$$\beta_k = -\frac{1}{2}\frac{\partial\chi^2}{\partial a_k} \quad \alpha_{kl} = \frac{1}{2}\frac{\partial^2\chi^2}{\partial a_k \partial a_l} \tag{A3.11}$$

making $[\alpha]=1/2\ \mathbf{D}$ in (A5.4), in terms of which that equation can be rewritten as the set of linear equations:

$$\sum_{l=1}^{M}\alpha_{kl}\delta_{al} = \beta_k \tag{A3.12}$$

This set is solved for the increments δa_l that, added to the current approximation, give the next approximation. In the context of least-squares, the matrix $[\alpha]$, equal to one-half times the Hessian matrix, is usually called the curvature matrix.

Eq. (A5.5), the steepest descent formula, translates to

$$\delta a_l = \text{constant} \times \beta_l \tag{A3.13}$$

Marquardt has put an elegant method, related to an earlier suggestion of Levenberg, for varying smoothly between the extremes of the inverse-Hessian method (A5.12) and the steepest descent method (A5.13). The latter method is used far from the minimum, switching continuously to the former as the minimum is approached. In this method (A5.13) is replaced by:

$$\delta a_l = \frac{1}{\lambda \alpha_{ll}} \beta_l \quad \text{or} \quad \lambda \alpha_{ll} \delta a_l = \beta_l \tag{A3.14}$$

Marquardt's second insight is that (A5.14) and (A5.12) can be combined if a new matrix α' is defined with the following characteristics:

$$\alpha'_{jj} = \alpha_{jj}(1+\lambda)$$

$$\alpha'_{jk} = \alpha_{jk} \qquad (j \neq k) \tag{A3.15}$$

and then (A5.14) and (A5.12) are replaced by:

$$\sum_{l=1}^{M} \alpha'_{kl} \delta_{al} = \beta_k \tag{A3.16}$$

When λ is very large, the matrix α' is forced into being diagonally dominant, so (A5.16) goes over to be identical to (A5.14). On the other hand, as λ approaches zero, (A5.16) goes over to (A5.12).

Given an initial value for the set of fitted parameters **a**, the Marquardt recipe is as follows:

- Compute χ^2 (**a**).
- Pick a modest value for λ, say λ=0.001.
- (*) Solve the linear equations (A5.16) for δ**a** and evaluate χ^2(**a**+δ**a**).
- If χ^2(**a**+δ**a**)$\geq\chi^2$(**a**), increase λ by a factor of 10 (or any substantial factor) and go back to (*).
- If χ^2(**a**+δ**a**)$< \chi^2$(**a**), decrease λ by a factor of 10, update the trial solution **a** $\leftarrow$ **a**+δ**a**, and go back to (*).

Iterating to convergence is generally wasteful and unnecessary since the minimum is at best only a statistical estimate of the parameters **a**.

Appendix 4

Linear Aerodynamic Coefficients

The values for the linear coefficients used in the ONERA Model are found from incompressible aerodynamics. Starting from the incompressible lift equation given by:

$$L = \rho\pi b^2\left[-\ddot{h} + U\dot{\theta} - ba\dot{\theta}\right] + 2\rho\pi bUC(K)\left[-\dot{h} + U\theta + b\left(\frac{1}{2} + a\right)\dot{\theta}\right] \tag{A4.1}$$

where a is the distance between the elastic axis and the mid-chord position. For this case, $a = \frac{1}{2}$ and h and θ are evaluated at the ¼ chord position. Replacing the time derivatives $(\dot{})$ and $(\ddot{})$ with dimensionless time (τ) where :

$$\tau = \left(\frac{Ut}{b}\right) \tag{A4.2}$$

gives the following:

$$l = \rho\pi U^2\left[-\dot{h} + b\dot{\theta} - ba\ddot{\theta}\right] + 2\rho\pi bU^2C(K)\left[-\frac{\dot{h}}{b} + \theta + \left(\frac{1}{2} - a\right)\dot{\theta}\right] \tag{A4.3}$$

From this equation and the definition of angle of attack given by $\alpha = \theta - \dot{h}/b$ and using $a = -1/2$ it is possible to derive the following:

$$L = \frac{1}{2}\rho U^2 c\left(C_{L_1} - C_{L_\gamma}\right) + \frac{1}{2}\rho U^2 c C_{L_\gamma} \tag{A4.4}$$

where

$$C_{L_1} - C_{L_\gamma} = -\pi\frac{\ddot{h}}{b} + \pi\dot{\theta} - \pi a\ddot{\theta} \tag{A4.5}$$

and

$$C_{L_\gamma} = C(K)\left[-2\pi\frac{\dot{h}}{b} + 2\pi\theta + 2\pi\left(\frac{1}{2} - a\right)\dot{\theta}\right] \tag{A4.6}$$

Using equation (A4.5) and the definitions of angle of attack and elastic axis gives:

$$C_{L_1} = \pi\dot{\alpha} + \frac{\pi}{2}\ddot{\theta} + C_{L_\gamma}$$

Which gives the linear coefficients $S_{L1} = \pi$, $S_{L2} = \pi/2$, and $S_{L3} = 0$.

Using Theodorsen's Function

$$F(K) = \frac{K^2\lambda_2 + \lambda_1^2}{\lambda_2^2 + K^2} \quad \text{and} \quad G(K) = \frac{K\lambda_1(\lambda_2 - 1)}{\lambda_1^2 + K^2} \tag{A4.7}$$

and the definition of angle of attack given earlier in equation (A4.6) results in the following:

$$\dot{C}_{L_\gamma} + 0.15C_{L_\gamma} = 0.15(2\pi)(\alpha + \dot{\theta}) + 0.55(2\pi)(\dot{\alpha} + \ddot{\theta}) \tag{A4.8}$$

From this, $\lambda_1 + 0.15$, $\lambda_2 + 0.55$, and $\alpha_{0_L} = 2\pi$. This procedure can be repeated for the moment coefficients in a similar manner.

Figures

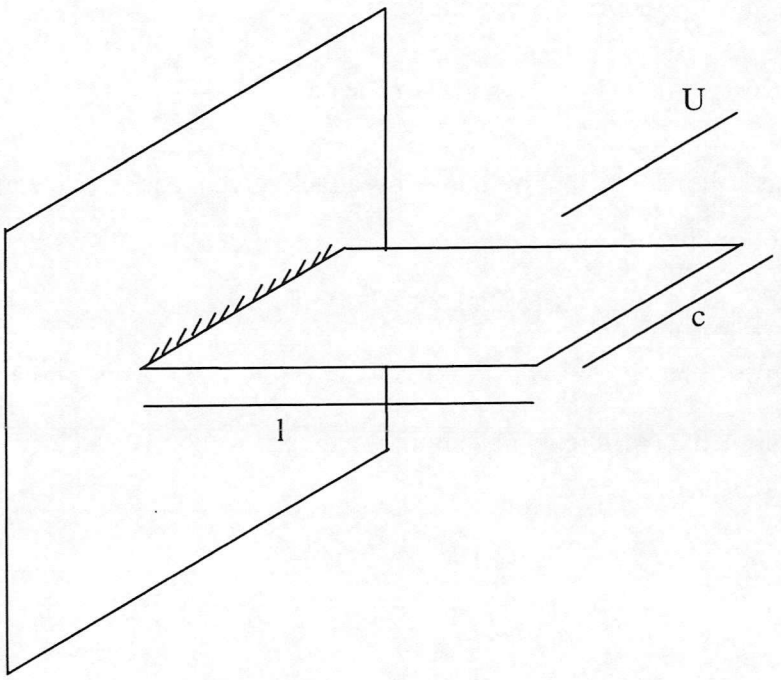


Fig. 1.1. Cantilever wing of uniform cross section.

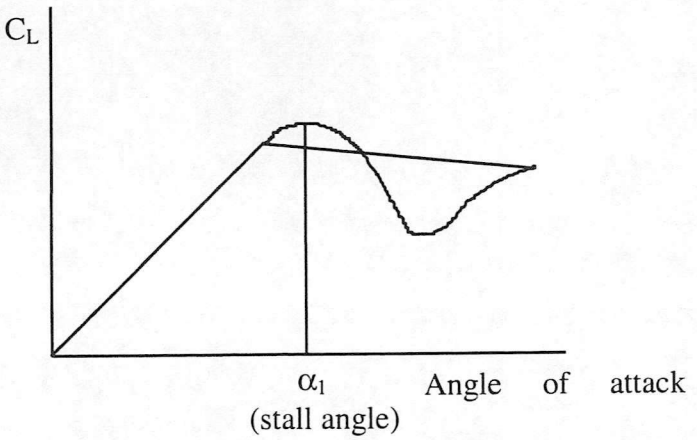


Fig.1.2. Variation of the lift coefficient (C_L) with the angle of attack.

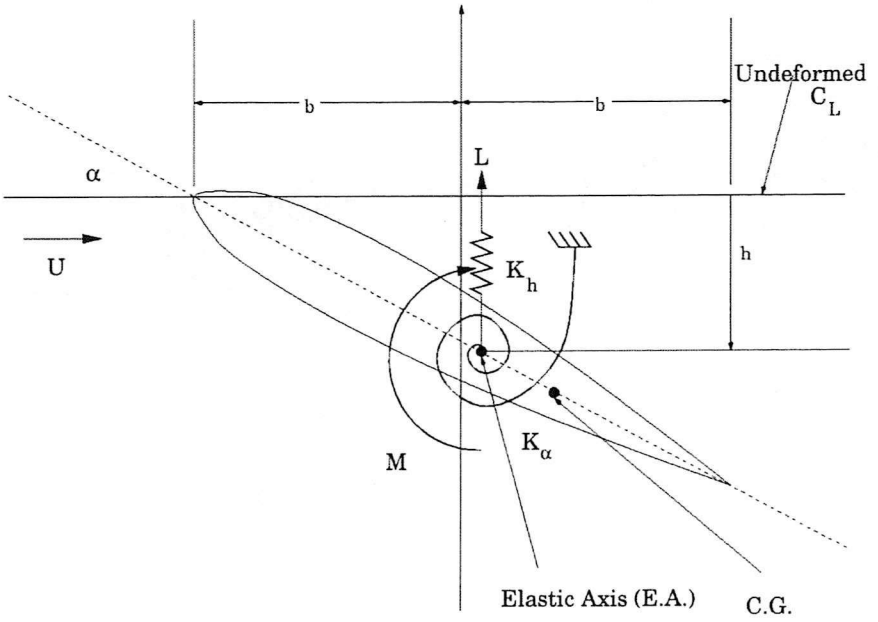
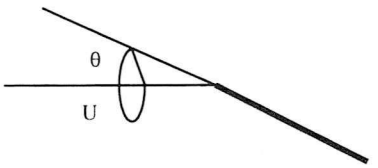
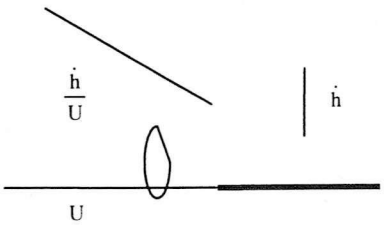


Fig.1.3. Two degree of freedom representation for an aerofoil.



a) Pitching



b) Plunging

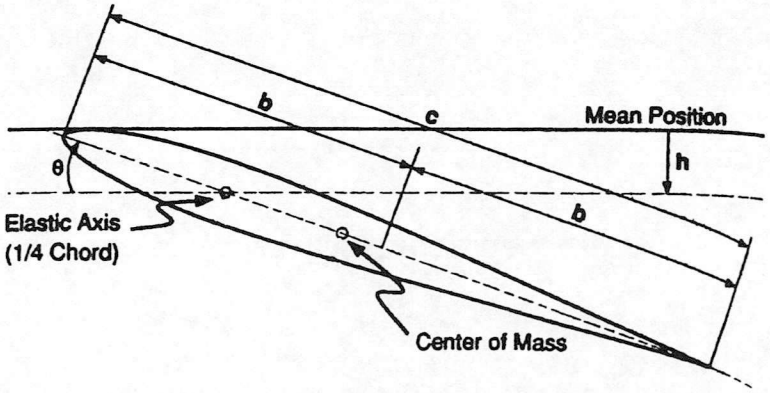


Fig.2.1. Definition of the variables in pitching and plunging motion.

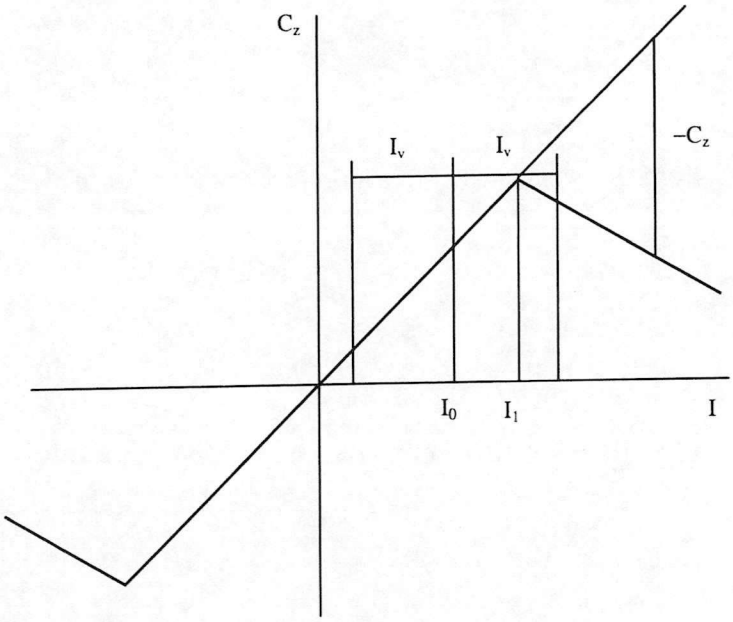


Fig.2.2. Single break-point approximation of the deviation $-C_z$.

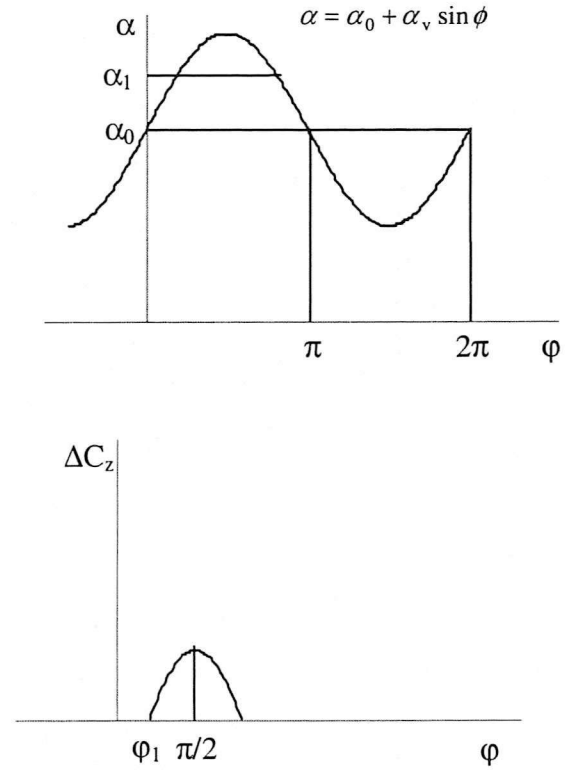


Fig. 2.3. Example of the oscillation over stall angle.

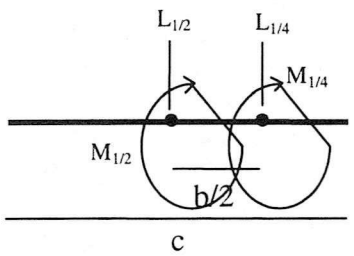


Fig. 2.4. Change of axis from 1/4 chord to 1/2 chord.

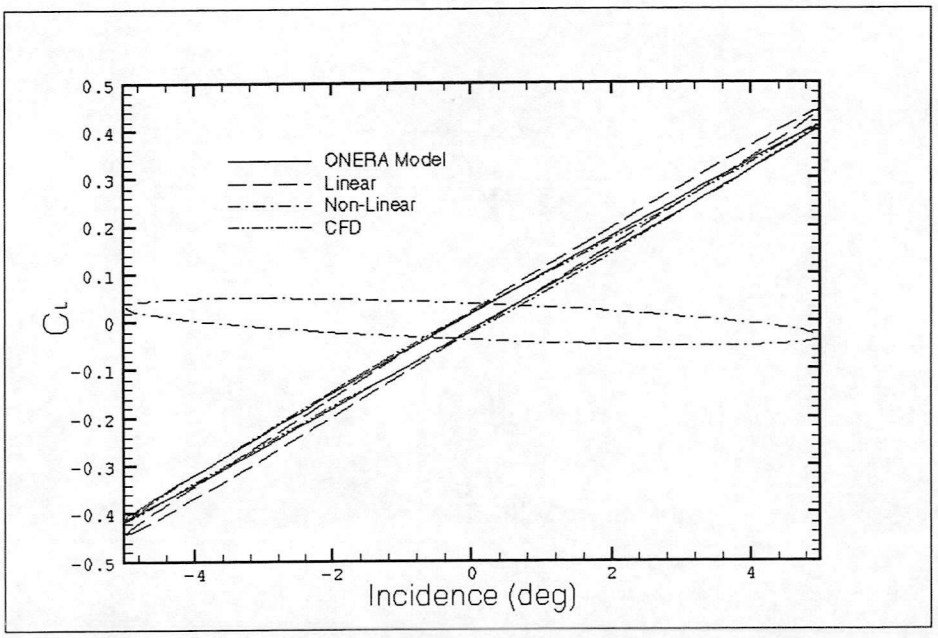


Fig.3.1. Comparison of the fitted and computed lift coefficients ($\alpha(t) = 5\sin(\omega t)$, M=0.302, Re=3.67x$10^6$, k-ω turbulence model).

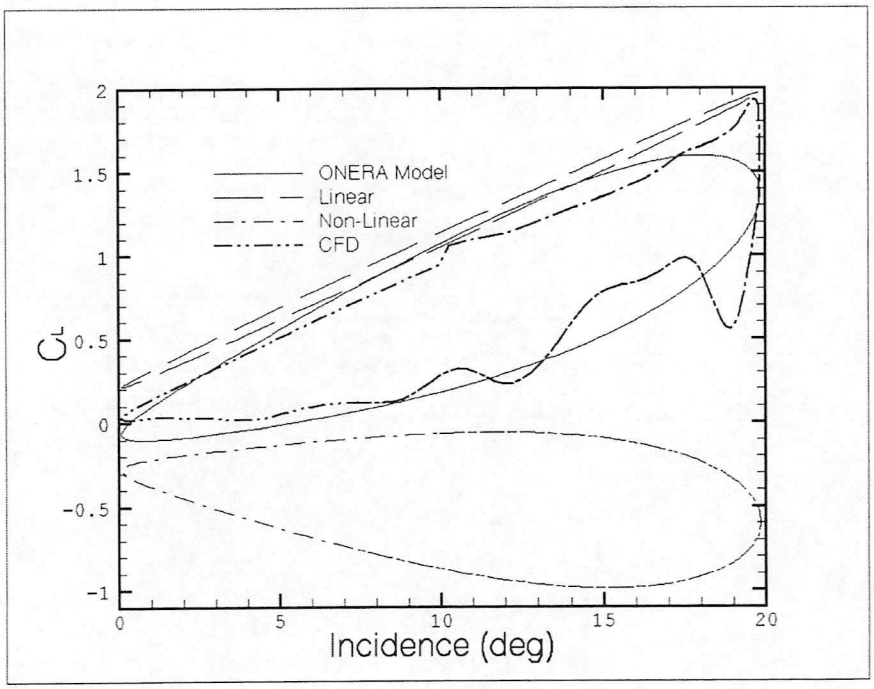


Fig.3.2. Comparison of the fitted and computed lift coefficients ($\alpha(t) = 9.97 + 9.88\sin(\omega t)$, M=0.302, Re=3.67x$10^6$, k-ω turbulence model).

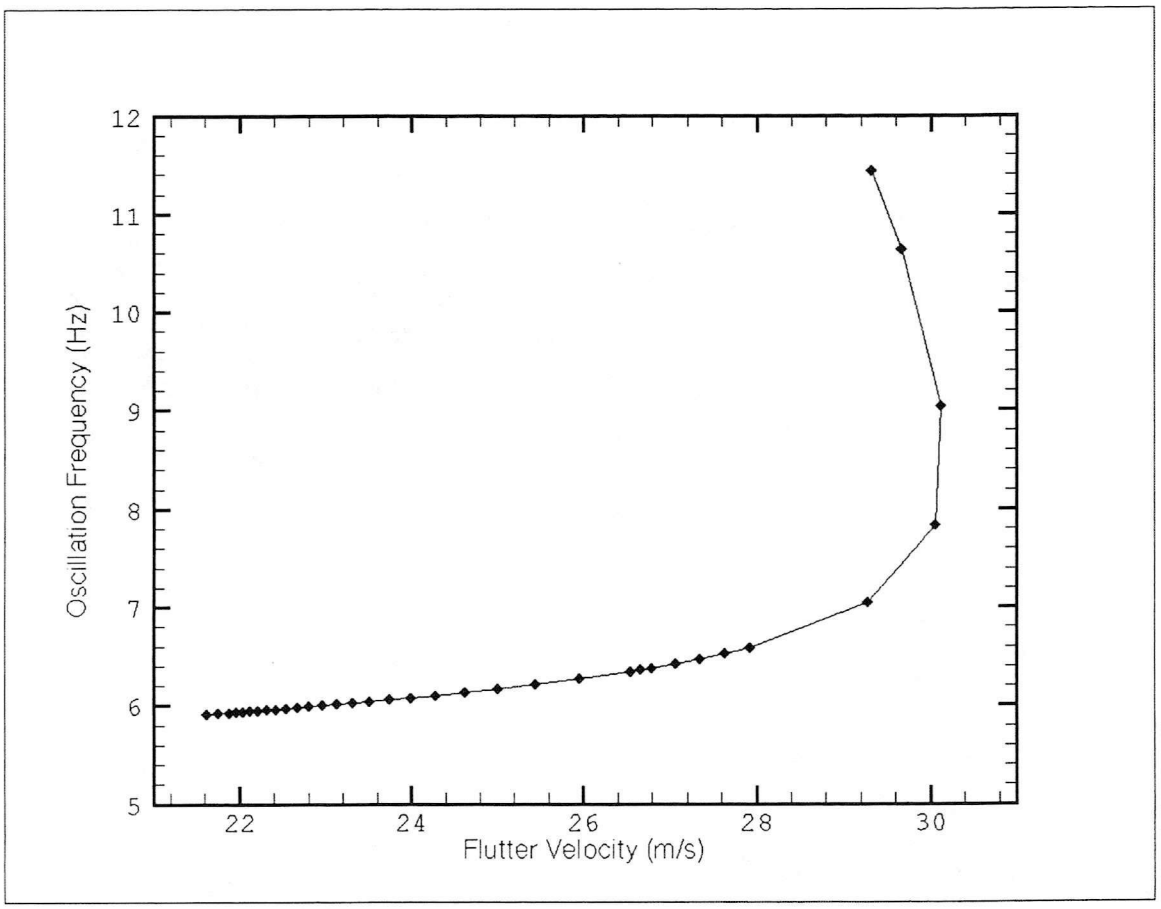


Fig.3.3. Prediction of flutter velocity and frequency using the ONERA aerodynamic model ($\alpha(t) = 9.97 + 9.88\sin(\omega t)$, M=0.302, Re=3.67x$10^6$, k-ω turbulence model).

Tables

Case	Mean angle of attack (deg.)	Amplitude of vibration (deg.)	Reduced frequency	Mach number	Reynolds number
1	0	5	0.145	0.302	3.67e6
2	9.97	9.88	0.145	0.302	3.67e6

Table 3.1. Description of the cases solved using the fluid solver.

Case	r_{10}	r_{12}	r_{20}	r_{22}	r_{30}	r_{32}
1	-9.1813	-19.4247	1.2356	1.2416	-6078.1	19.3940
2	-5.6958	-17.4499	0.6425	1.6833	-0.8193	16.3939
Exper. Data	-3.8187	-8.1573	0.5241	1.0171	-0.6695	17.1221

Table 3.2. Parameters r_{10}, r_{12}, r_{20}, r_{22}, r_{30} and r_{32} obtained with the fitting.

Case	C_{L0}	C_{Ls1}	C_{Lc1}	C_{Ls2}	C_{Lc2}
1	0.0872	0.0166	0.3183	-0.0226	-0.2177
2	0.7335	-0.1923	-0.1663	0.3535	0.2122
Exper. Data	0.7730	-0.1924	-0.1445	0.3563	0.1934

Table 3.3. Harmonic decomposition of the computed and experimental lift coefficient curves.

Mass (kg/m)(M)	0.283
Moment of inertia (kg m)(I_a)	343e-6
Density (kg/m^3)(ρ)	1.23
Length (m)(l)*	0.33
Chord (m)(c)*	0.076
Stall angle (deg)(α_1)	9
Bending frequency (rad/s)(ω_h)	27.02
Torsion frequency (rad/s)(ω_a)	154.6

Table 3.4. Parameters and constants used in the flutter program. (This values are from [3.1] except the ones marked with * which are from [1.8].)

Guide to computer programs

1. Introduction

Two different programs were used in the project, apart from the CFD code: the fitting program and the flutter one. The fitting program is based on the ONERA model, and tries to fit that model to a given lift coefficient loop in order to obtain the r_i coefficients of the non-linear equation of the ONERA model. It works following the Levenberg-Marquardt method. The flutter program calculates, using the ONERA model for the aerodynamic part, the flutter velocity and frequency for a given conditions on an aerofoil.

2. Fitting program

The variables that the program needs are introduced in a file called 'aeroel2.in' and are summarised in the following table. Apart from those variables, the program also needs a file with the values of the lift coefficient, which are to be fitted to. This file is named 'lift.dat'. It contains two columns: the first column is the angle of attack and the second one the lift coefficient. These values can come from experiments or from a CFD code.

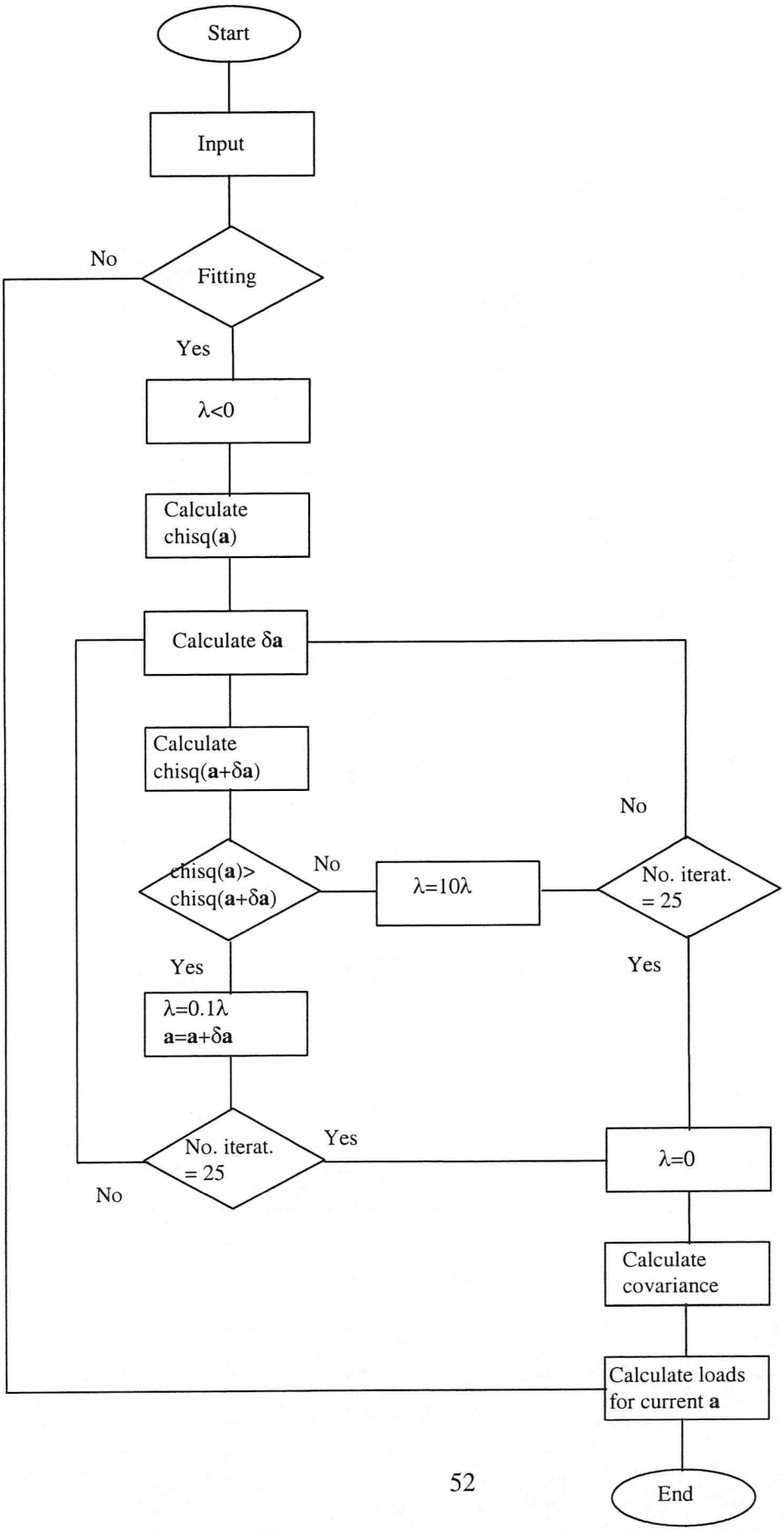
The output of the program is written in the file 'aeroel2.out'. It contains the best-fit values for the parameters r_{10}, r_{20}, r_{30}, r_{12}, r_{22} and r_{32} and the value of chi-square, which measures the agreement between the input file and the calculated coefficient, followed by the lift or moment coefficient calculated using those parameters for every angle of attack of the input file ('lift.dat').

The program was used in the following way: after obtaining a set of values for the lift coefficient using the flow solver (it could be obtained by means of experiments), these values are introduced in the 'lift.dat' file. The characteristics of the case are introduced in the 'aeroel2.in' file. The parameter NFIT is set to 1, in order to carry out the fitting, and then the program is run. The r_i parameters and the calculated lift coefficient are available in the 'aeroel2.out' file. Once the parameters are estimated for the lift coefficient, the

moment coefficient is calculated using those same parameters. This is done by setting the NFIT parameter of the 'aeroel2.in' file to 0, and running the program. Now, the fitting is not done, but the r_i parameters estimated before are specified in the input file and used to calculate the moment coefficient. Obviously, some of the input variables has to be changed to calculate the moment coefficient, such as S_{zi} or b_1.

Input	Symbol in the dissertation	Definition
K	k	Reduced frequency
ALPHA0D	α_0	Average wing angle (deg)
ALPHAVD	α_v	Amplitude of vibration angle (deg)
B	b	Semi-chord length (m)
SL1,SL2,SL3	S_{z1}, S_{z2}, S_{z3}	Coefficients in linear equations ONERA model
AOL	a_{0L}, a_{0M}	Linear lift curve slope
LAM1,LAM2	λ_1, λ_2	Coefficients in linear equations ONERA model
THC	θ_c	Cosine component of pitch angle
HC	h_c	Cosine component of the 1/4 chord deflection
HS	h_s	Sine component of the 1/4 chord deflection
R10,R20,R30R 12,R22,R32	r_{10}, r_{20}, r_{30}, r_{12}, r_{22}, r_{32}	Initial guess of the Ri coefficients in the non-linear equations of the ONERA model
ALPHA1	α_1	Stall angle (deg)
B1	b_1	Non-linear slope of the deviation from linear force curve.
NFIT	No reference in the dissertation	0 if no fitting (calculate the loads using the current values of R10, R20, R30, R12, R22, R32, 1 if fitting

The working of the program is summarised in the following flow chart.


Start
Input
Fitting
No
Yes
λ<0
Calculate chisq(a)
Calculate δa
Calculate chisq(a+δa)
chisq(a)> chisq(a+δa)
No
λ=10λ
No. iterat. = 25
No
Yes
λ=0.1λ
a=a+δa
No. iterat. = 25
Yes
No
λ=0
Calculate covariance
Calculate loads for current a
End

3. Flutter program

The input of the program is introduced in a file named 'aeroel3.in'. It contains the following variables:

Input	Symbol in the dissertation	Definition
R10	r_{12}	Coefficients of ONERA non-linear aerodynamic equations
R20	r_{20}	
R30	r_{30}	
R12	r_{12}	
R22	r_{22}	
R32	r_{32}	
M	M	Total mass per unit length
IA	I_a	Mass moment of inertia per unit length
RHO	ρ	Wing material density
C	c	Chord length
L	l	Wing length
WH	ω_h	Uncoupled torsion frequency
WA	ω_a	Uncoupled bending frequency
ALPHA1D	α_1	Stall angle
THETA0D	θ_R	Root angle
X	x	State vector in Newton-Raphson scheme
Q2S	Q_{2s}	Sine component of second mode of vibration

For every Q2S, the program calculates the flutter velocity and frequency. It uses the Newton-Raphson method to solve the system of six equations (2.130-2.135 in the dissertation, see also Appendix 2), deduced from the aeroelastic model. Those equations are formulated in the FUNCV subroutine in the form:

$F_i(x_1,...x_6)=0$ (FVEC or P in the program)

The vector $(x_1,...x_6)$ is called state vector. It includes q_{10} (mean component of first mode, bending), q_{20} (mean component of second mode, torsion), q_{1s} (sine component of first mode), q_{1c} (cosine component of first mode), k_a (reduced torsional frequency, k2 in the program) and k (reduced frequency).

The Newton-Raphson method is used:

$$F_i(x+\delta x) = F_i(x) + \sum_{j=1}^{6} \frac{\partial F_i}{\partial x_j} \cdot \delta x_j + H.O.T. = 0$$

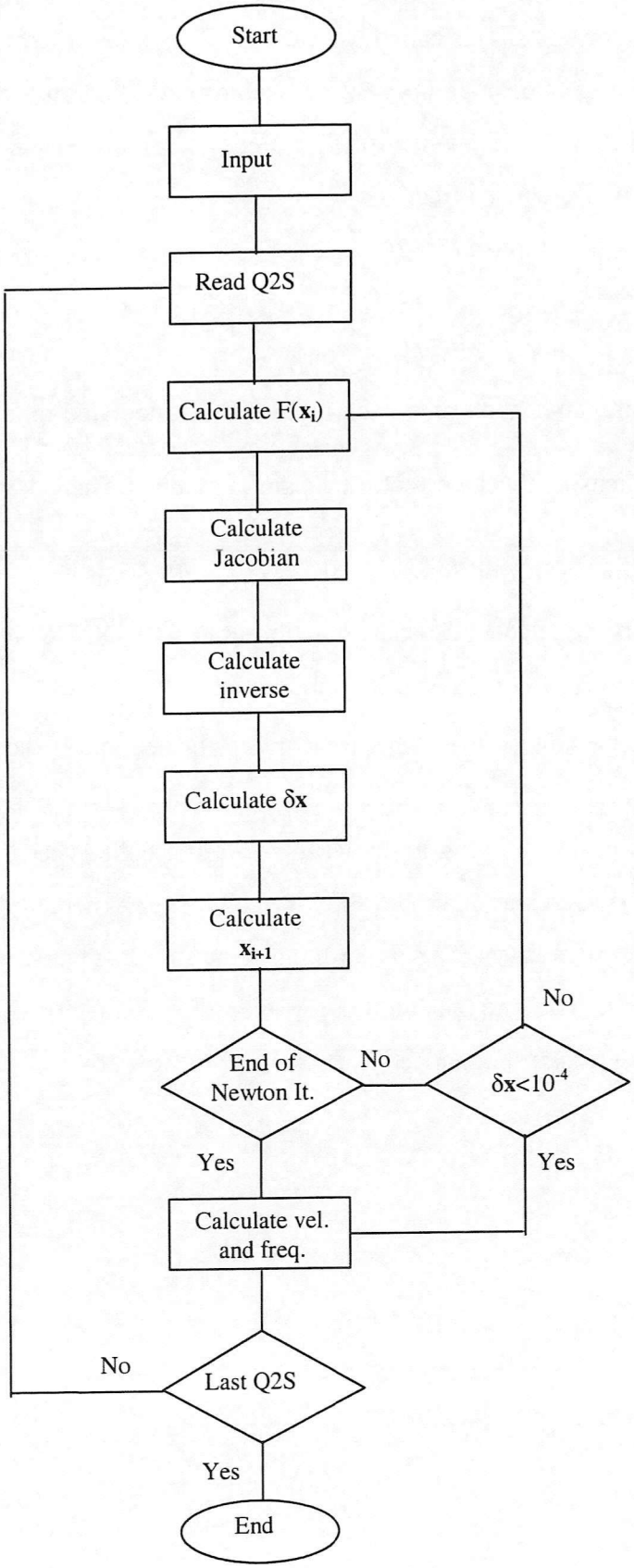
The summation is the Jacobian matrix. It is calculated numerically in the FDJAC subroutine, increasing the state vector by a small number. Then $J = \frac{\partial F}{\partial x_j} = \frac{\Delta F}{\Delta x_j}$.

Thus, the correction is: $\delta x = -J^{-1} \cdot F(x)$, at the nth iteration. The inverse of the Jacobian matrix is calculated using the subroutines LUDCMP and LUBKSB. And the estimation of the vector at the next iteration is: $x_{n+1} = x_n + \delta x$.

For the present work 25 iterations are performed and if the change in variables is not less than 10^{-4} the iteration procedure is considered as divergent.

The output of the program is written to a file called 'aeroel3.out'. It contains, in columns, Q2S (which was an input), the x vector (which is actually q10, q20, q1s, q1c, k2 and k), the flutter velocity (FS in the program), the flutter frequency (FQ), the mean angle of attack (ALPHA0D), and the amplitude of vibration (AVIBD).

The program is summarised in the following flow chart.


Start
Input
Read Q2S
Calculate F(x_i)
Calculate Jacobian
Calculate inverse
Calculate δx
Calculate x_{i+1}
End of Newton It.
No
δx<10^-4
No
Yes
Yes
Calculate vel. and freq.
No
Last Q2S
Yes
End

4. Listings

4.1. Fitting program

```
      PROGRAM FIT
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     FITTING PROGRAM TO ESTIMATE THE Ri PARAMETERS OF THE
C     NON-LINEAR EQUATION OF THE ONERA MODEL
C
C     INPUT: 'lift.dat': CONTAINS THE LIFT COEFFICIENT LOOP TO FIT TO
C
C     K             REDUCED FREQUENCY
C     ALPHA0D       AVERAGE WING ANGLE (DEG)
C     ALPHAVD       VIBRATION ANGLE (DEG)
C     B             SEMICHORD LENGTH (M)
C     SL1,SL2,SL3   COEFFICIENTS IN LINEAR EQUATIONS ONERA MODEL
C     AOL           LINEAR LIFT CURVE SLOPE
C     LAM1,LAM2     COEFFICIENTS IN LINEAR EQUATIONS ONERA MODEL
C     THC           COSINE COMPONENT OF THE PITCH ANGLE
C     HC            COSINE COMPONENT OF THE 1/4 CHORD DEFLECTION
C     HS            SINE COMPONENT OF THE 1/4 CHORD DEFLECTION
C     R10,R20,R30,  INITIAL GUESS OF THE Ri COEFFICIENTS
C     R12,R22,R32   IN THE NON-LINEAR EQUATIONS OF THE ONERA MODEL
C     ALPHA1        STALL ANGLE (DEG)
C     B1            NON-LINEAR SLOPE OF THE DEVIATION FROM LINEAR FORCE CURVE
C     NFIT          0 FOR NO FITTING (CALCULATES THE LOADS USING THE CURRENT
C                   VALUES OF R10,R20,R30,R12,R22,R32, 1 IF FITTING IS REQUIRED
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      DIMENSION EXPERZ(500),EXPERA(500),SIG(500)
      DIMENSION APAR(6),LISTA(6),COVAR(500,500),ALPHA(500,500)
      EXTERNAL FLIFT
C
      REAL  B,SL1,SL2,SL3,AOL,LAM1,LAM2,THC,HC,HS,ALPHA1,B1,K,
     F      ALPHA0D,ALPHAVD,R10,R20,R30,R12,R22,R32,Z
      COMMON/L/ B,SL1,SL2,SL3,AOL,LAM1,LAM2,THC,HC,HS
      COMMON/NL/ ALPHA1,B1
      COMMON/FUN/ K,ALPHA0D,ALPHAVD
      NA=6
C
C     READ THE FILE CONTAINING THE VALUES TO FIT TO
C     NEXPER     NUMBER OF POINTS TO FIT
C     EXPERA(I)  ANGLE OF ATTACK (DEG)
C     EXPERZ(I)  LIFT COEFFICIENT
C
      I=1
      OPEN (11,FILE='lift.dat',STATUS='OLD')
CG222   READ (11,*,END=333) a1,a2,a3, EXPERA(I),EXPERZ(I)
222   READ (11,*,END=333) EXPERA(I),EXPERZ(I)
      I=I+1
      GOTO 222
333   CLOSE(11)
      NEXPER=I-1
C
C     READ THE VARIABLES TO BE USED IN THE PROGRAM
C
      OPEN(10,FILE='aeroel2.in',STATUS='OLD')
      READ (10,7000) K,ALPHA0D,ALPHAVD,B,SL1,SL2,SL3,AOL
      READ (10,7000) LAM1,LAM2,THC,HC,HS,R10,R12,R20
      READ (10,7000) R22,R30,R32,ALPHA1,B1
      READ (10,7000) LISTA(1),LISTA(2),LISTA(3),
     F               LISTA(4),LISTA(5),LISTA(6)
      READ (10,*) NFIT
      READ (10,*) NITS
C
C     INITIAL GUESSES FOR THE Ri PARAMETERS ARE PUT INTO APAR.
C
      APAR(1)=R10
```

```
      APAR(2)=R20
      APAR(3)=R30
      APAR(4)=R12
      APAR(5)=R22
      APAR(6)=R32
C
C     CREATES THE OUTPUT FILE
C
      OPEN(9,FILE='aeroel2.out',STATUS='UNKNOWN')
C
C     IF NFIT.EQ.0 FITTING IS NOT DONE, GOES DIRECTLY TO THE CALCULATION
C     USING THE CURRENT Ri. ELSE, THE FITTING IS DONE.
C
      IF (NFIT.EQ.0) GOTO 2
C
C     FITTING BEGINS.
C
      NCA=500
      DO I=1,NEXPER
         SIG(I)=1.
      ENDDO
      ALPHA1=ALPHA1*3.141593/180.0
C
C     INITIAL CALL TO MRQMIN. ALAMBDA IS RETURNED ON THE FIRST AND ALL
C     SUBSEQUENT CALLS AS THE SUGGESTED VALUE OF LAMBDA FOR THE NEXT
C     ITERATION; A IS ALWAYS RETURNED AS THE BEST PARAMETERS FOUND SO FAR.
C
      ALAMDA=-1
      CALL MRQMIN(EXPERA,EXPERZ,SIG,NEXPER,APAR,LISTA,6,
     F            COVAR,ALPHA,NCA,CHISQ,FLIFT,ALAMDA)
C
C     CALL MRQMIN NITS TIMES, ENOUGH TO ACHIEVE GOOD RESULTS.
C
      NCON=0
111   CONTINUE
      CALL MRQMIN(EXPERA,EXPERZ,SIG,NEXPER,APAR,LISTA,6,
     F            COVAR,ALPHA,NCA,CHISQ,FLIFT,ALAMDA)
      NCON=NCON+1
      IF (NCON.LT.NITS) GOTO 111
C
C     ALAMDA IS SET TO ZERO BEFORE THE FINAL CALL TO MRQMIN
C
      ALAMDA=0.0
      CALL MRQMIN(EXPERA,EXPERZ,SIG,NEXPER,APAR,LISTA,6,
     F            COVAR,ALPHA,NCA,CHISQ,FLIFT,ALAMDA)
C
C     OUTPUT WRITTEN IN THE OUTPUT FILE. APAR(i) CORRESPONDS TO Ri
C
      WRITE(9,8000) APAR(1),APAR(2),APAR(3),APAR(4),APAR(5),APAR(6)
      WRITE(9,8001) CHISQ
C
C     USING THE ESTIMATED Ri, THE LIFT OR MOMENT COEFFICIENT IS
C     CALCULATED AND WRITTEN TO 'aeroel2.out', FOR EVERY ANGLE
C     OF ATTACK OF THE 'lift.dat' FILE.
C
2     DO L=1,NEXPER
         NF=1
         IF (L.EQ.1) THEN
CGSOS          CALL FLIFT(EXPERA(L),-13.,APAR,NF,Z,DYDA,NA)
            CALL FLIFT(EXPERA(L),EXPERA(NEXPER),APAR,NF,Z,DYDA,NA)
         ELSE
            CALL FLIFT(EXPERA(L),EXPERA(L-1),APAR,NF,Z,DYDA,NA)
         ENDIF
      ENDDO
      STOP
C
C     FORMAT STATEMENTS
C
7000  FORMAT(1X,F10.5)
8000  FORMAT(1X,'#',6(1X,F10.4))
8001  FORMAT(1X,'#', (1X,F10.4))
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE FUNCVL(ALPHA0,AVIB,K,PHI,THETA,CLGO,PLS,PLC,F,G,P)
C     CALCULATE LINEAR LIFT. P IS CL1 IN THE TEXT OF THE DISSERTATION
C     PLS AND PLC ARE THE SIN AND COS COMPONENTS OF THE LINEAR
C     CONTRIBUTION TO THE GENERAL AERODYNAMIC FORCE COEFFICIENT. REST
C     OF THE VARIABLES FOLLOW THE NOMENCLATURE OF THE DISSERTATION.
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL SL1,SL2,SL3,AOL,LAM1,LAM2,ALPHA0,AVIB,
     F     THC,P,PHI,K,HC,HS,B,THETA,PLS,PLC,LS,LC,F,G,CLGO,CLGS,CLGC
      COMMON/L/ B,SL1,SL2,SL3,AOL,LAM1,LAM2,THC,HC,HS
C
      LS=AOL*(AVIB-K*HC/B-K*THC)
      LC=AOL*(THC+K*HS/B+K*AVIB)
      F=(K*K*LAM2+LAM1*LAM1)/(LAM1*LAM1+K*K)
      G=K*LAM1*(LAM2-1)/(LAM1*LAM1+K*K)
      CLGO=AOL*ALPHA0
      CLGS=F*LS-G*LC
      CLGC=G*LS+F*LC
      THETA=ALPHA0+AVIB*SIN(PHI)+THC*COS(PHI)
      PLS=SL1*(-K*THC-(HS/B)*K*K)-SL2*AVIB*K*K-SL3*K*THC +CLGS
      PLC=SL1*(AVIB*K-(HC/B)*K*K)-SL2* THC*K*K+SL3*K*AVIB+CLGC
      P=CLGO+PLS*SIN(PHI)+PLC*COS(PHI)
C
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE FUNCVN(A,NA,ALPHA0,AVIB,R1,R2,R3,K,PHI,CZ2C2,CZ2S2,
     F                  CZ2C1,CZ2S1,CZ20,P2,
     F D_P2_DR10,D_P2_DR20,D_P2_DR30,D_P2_DR12,D_P2_DR22,D_P2_DR32)
C     CALCULATE NON-LINEAR PART OF THE LIFT. P2 IS THE NON-LINEAR
C     CONTRIBUTION TO THE GENERAL AERODYNAMIC LIFT COEFFICIENT.
C     DERIVATIVES OF P2 WITH RESPECT TO THE PARAMETERS Ri ARE
C     CALCULATED. THEY ARE NECESSARY TO DO THE FITTING.
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL   A(NA),DCZ0,DCZS1,DCZC2,R0,RS1,RC1,RS2,K1,K2,K3,K4,CZ2C2,
     F       CZ2S2,CZ2C1,CZ2S1,CZ20,ALPHA1,B1,ALPHA0,AVIB,R1,R2,R3,
     F       PHI1,K,P2,PHI,PENT2,PHI1T,R10,R20,R30,R12,R22,R32,ALPHA1T
      COMMON/NL/ ALPHA1,B1
C
      R10=A(1)
      R20=A(2)
      R30=A(3)
      R12=A(4)
      R22=A(5)
      R32=A(6)
      ALPHA1T=-ALPHA1
      PENT=(ALPHA1-ALPHA0)/AVIB
      IF (PENT.GT.1.) THEN
         PHI1=1.571
      ELSE IF (PENT.LT.-1.) THEN
         PHI1=-1.571
      ELSE
         PHI1=ASIN(PENT)
      ENDIF
      PENT2=(ALPHA1T-ALPHA0)/AVIB
      IF (PENT2.GT.1.) THEN
         PHI1T=1.571
      ELSE IF (PENT2.LT.-1.) THEN
         PHI1T=-1.571
      ELSE
         PHI1T=ASIN(PENT2)
      ENDIF
C
      DCZ0 =(B1*AVIB/3.14159)*(-PENT*(1.571-PHI1 )+COS(PHI1 ))
     F     -(B1*AVIB/3.14159)*(PENT2*(1.571+PHI1T)+COS(PHI1T))
      DCZS1=(B1*AVIB/3.14159)*( (1.571-PHI1 )-0.5*SIN(2.*PHI1 ))
     F     -(B1*AVIB/3.14159)*(-(1.571+PHI1T)-0.5*SIN(2.*PHI1T))
      DCZC2=(B1*AVIB/3.14159)*(-.5*COS(PHI1 )-0.166667*COS(3.*PHI1 ))
     F     -(B1*AVIB/3.14159)*(-.5*COS(PHI1T)-0.166667*COS(3.*PHI1T))
      R1=R10+R12*DCZ0*DCZ0
      R2=(R20+R22*DCZ0*DCZ0)**2
      R3=(R30+R32*DCZ0*DCZ0)*R20
      K1=R2-K*K
      K2=R1*K
      K3=R2-4*K*K
      K4=2*R1*K
      R0=-R2*DCZ0
      RS1=-R2*DCZS1
      RC1=-R3*K*DCZS1
      RS2=2*R3*K*DCZC2
      RC2=-R2*DCZC2
      CZ20=R0/R2
      CZ2S1=(K1*RS1+K2*RC1)/(K1*K1+K2*K2)
      CZ2C1=(K1*RC1-K2*RS1)/(K1*K1+K2*K2)
```

```
      CZ2S2=(K3*RS2+K4*RC2)/(K3*K3+K4*K4)
      CZ2C2=(K3*RC2-K4*RS2)/(K3*K3+K4*K4)
      P2=CZ20+CZ2S1*SIN(    PHI)+CZ2C1*COS(    PHI)
     F       +CZ2S2*SIN(2.*PHI)+CZ2C2*COS(2.*PHI)
C
C     DERIVATIVES
C
      D_R1_DR10=1.
      D_R1_DR20=0.
      D_R1_DR30=0.
      D_R1_DR12=DCZ0*DCZ0
      D_R1_DR22=0.
      D_R1_DR32=0.
C
      D_R2_DR10=0.
      D_R2_DR20=2.*(R20+R22*DCZ0*DCZ0)
      D_R2_DR30=0.
      D_R2_DR12=0.
      D_R2_DR22=2.*(R20+R22*DCZ0*DCZ0)*DCZ0*DCZ0
      D_R2_DR32=0.
C
      D_R3_DR10=0.
      D_R3_DR20=R30+R32*DCZ0*DCZ0
      D_R3_DR30=R20
      D_R3_DR12=0.
      D_R3_DR22=0.
      D_R3_DR32=DCZ0*DCZ0*R20
C
      D_R0_DR10=-DCZ0*D_R2_DR10
      D_R0_DR20=-DCZ0*D_R2_DR20
      D_R0_DR30=-DCZ0*D_R2_DR30
      D_R0_DR12=-DCZ0*D_R2_DR12
      D_R0_DR22=-DCZ0*D_R2_DR22
      D_R0_DR32=-DCZ0*D_R2_DR32
C
      D_RS2_DR10=2.*K*DCZC2*D_R3_DR10
      D_RS2_DR20=2.*K*DCZC2*D_R3_DR20
      D_RS2_DR30=2.*K*DCZC2*D_R3_DR30
      D_RS2_DR12=2.*K*DCZC2*D_R3_DR12
      D_RS2_DR22=2.*K*DCZC2*D_R3_DR22
      D_RS2_DR32=2.*K*DCZC2*D_R3_DR32
C
      D_RC2_DR10=-DCZC2*D_R2_DR10
      D_RC2_DR20=-DCZC2*D_R2_DR20
      D_RC2_DR30=-DCZC2*D_R2_DR30
      D_RC2_DR12=-DCZC2*D_R2_DR12
      D_RC2_DR22=-DCZC2*D_R2_DR22
      D_RC2_DR32=-DCZC2*D_R2_DR32
C
      D_RC1_DR10=-K*DCZS1*D_R3_DR10
      D_RC1_DR20=-K*DCZS1*D_R3_DR20
      D_RC1_DR30=-K*DCZS1*D_R3_DR30
      D_RC1_DR12=-K*DCZS1*D_R3_DR12
      D_RC1_DR22=-K*DCZS1*D_R3_DR22
      D_RC1_DR32=-K*DCZS1*D_R3_DR32
C
      D_RS1_DR10=-DCZS1*D_R2_DR10
      D_RS1_DR20=-DCZS1*D_R2_DR20
      D_RS1_DR30=-DCZS1*D_R2_DR30
      D_RS1_DR12=-DCZS1*D_R2_DR12
      D_RS1_DR22=-DCZS1*D_R2_DR22
      D_RS1_DR32=-DCZS1*D_R2_DR32
C
      D_K1_DR10=D_R2_DR10
      D_K1_DR20=D_R2_DR20
      D_K1_DR30=D_R2_DR30
      D_K1_DR12=D_R2_DR12
      D_K1_DR22=D_R2_DR22
      D_K1_DR32=D_R2_DR32
C
      D_K2_DR10=K*D_R1_DR10
      D_K2_DR20=K*D_R1_DR20
      D_K2_DR30=K*D_R1_DR30
      D_K2_DR12=K*D_R1_DR12
      D_K2_DR22=K*D_R1_DR22
      D_K2_DR32=K*D_R1_DR32
C
      D_K3_DR10=D_R2_DR10
```

```
      D_K3_DR20=D_R2_DR20
      D_K3_DR30=D_R2_DR30
      D_K3_DR12=D_R2_DR12
      D_K3_DR22=D_R2_DR22
      D_K3_DR32=D_R2_DR32
C
      D_K4_DR10=2*K*D_R1_DR10
      D_K4_DR20=2*K*D_R1_DR20
      D_K4_DR30=2*K*D_R1_DR30
      D_K4_DR12=2*K*D_R1_DR12
      D_K4_DR22=2*K*D_R1_DR22
      D_K4_DR32=2*K*D_R1_DR32
C
      D_CZ2S1_DR10=(D_K1_DR10*RS1+K1*D_RS1_DR10+D_K2_DR10*RC1+
     F                           K2*D_RC1_DR10)/(K1*K1+K2*K2)-
     F             (K1*RS1+K2*RC1)*(2.*K1*D_K1_DR10+2.*K2*D_K2_DR10)
     F             /(K1*K1+K2*K2)
      D_CZ2S1_DR20=(D_K1_DR20*RS1+K1*D_RS1_DR20+D_K2_DR20*RC1+
     F                           K2*D_RC1_DR20)/(K1*K1+K2*K2)-
     F             (K1*RS1+K2*RC1)*(2.*K1*D_K1_DR20+2.*K2*D_K2_DR20)
     F             /(K1*K1+K2*K2)
      D_CZ2S1_DR30=(D_K1_DR30*RS1+K1*D_RS1_DR30+D_K2_DR30*RC1+
     F                           K2*D_RC1_DR30)/(K1*K1+K2*K2)-
     F             (K1*RS1+K2*RC1)*(2.*K1*D_K1_DR30+2.*K2*D_K2_DR30)
     F             /(K1*K1+K2*K2)
      D_CZ2S1_DR12=(D_K1_DR12*RS1+K1*D_RS1_DR12+D_K2_DR12*RC1+
     F                           K2*D_RC1_DR12)/(K1*K1+K2*K2)-
     F             (K1*RS1+K2*RC1)*(2.*K1*D_K1_DR12+2.*K2*D_K2_DR12)
     F             /(K1*K1+K2*K2)
      D_CZ2S1_DR22=(D_K1_DR22*RS1+K1*D_RS1_DR22+D_K2_DR22*RC1+
     F                           K2*D_RC1_DR22)/(K1*K1+K2*K2)-
     F             (K1*RS1+K2*RC1)*(2.*K1*D_K1_DR22+2.*K2*D_K2_DR22)
     F             /(K1*K1+K2*K2)
      D_CZ2S1_DR32=(D_K1_DR32*RS1+K1*D_RS1_DR32+D_K2_DR32*RC1+
     F                           K2*D_RC1_DR32)/(K1*K1+K2*K2)-
     F             (K1*RS1+K2*RC1)*(2.*K1*D_K1_DR32+2.*K2*D_K2_DR32)
     F             /(K1*K1+K2*K2)
C
      D_CZ2C1_DR10=(D_K1_DR10*RC1+K1*D_RC1_DR10-D_K2_DR10*RS1-
     F                           K2*D_RS1_DR10)/(K1*K1+K2*K2)-
     F             (K1*RC1-K2*RS1)*(2.*K1*D_K1_DR10+2.*K2*D_K2_DR10)
     F             /(K1*K1+K2*K2)
      D_CZ2C1_DR20=(D_K1_DR20*RC1+K1*D_RC1_DR20-D_K2_DR20*RS1-
     F                           K2*D_RS1_DR20)/(K1*K1+K2*K2)-
     F             (K1*RC1-K2*RS1)*(2.*K1*D_K1_DR20+2.*K2*D_K2_DR20)
     F             /(K1*K1+K2*K2)
      D_CZ2C1_DR30=(D_K1_DR30*RC1+K1*D_RC1_DR30-D_K2_DR30*RS1-
     F                           K2*D_RS1_DR30)/(K1*K1+K2*K2)-
     F             (K1*RC1-K2*RS1)*(2.*K1*D_K1_DR30+2.*K2*D_K2_DR30)
     F             /(K1*K1+K2*K2)
      D_CZ2C1_DR12=(D_K1_DR12*RC1+K1*D_RC1_DR12-D_K2_DR12*RS1-
     F                           K2*D_RS1_DR12)/(K1*K1+K2*K2)-
     F             (K1*RC1-K2*RS1)*(2.*K1*D_K1_DR12+2.*K2*D_K2_DR12)
     F             /(K1*K1+K2*K2)
      D_CZ2C1_DR22=(D_K1_DR22*RC1+K1*D_RC1_DR22-D_K2_DR22*RS1-
     F                           K2*D_RS1_DR22)/(K1*K1+K2*K2)-
     F             (K1*RC1-K2*RS1)*(2.*K1*D_K1_DR22+2.*K2*D_K2_DR22)
     F             /(K1*K1+K2*K2)
      D_CZ2C1_DR32=(D_K1_DR32*RC1+K1*D_RC1_DR32-D_K2_DR32*RS1-
     F                           K2*D_RS1_DR32)/(K1*K1+K2*K2)-
     F             (K1*RC1-K2*RS1)*(2.*K1*D_K1_DR32+2.*K2*D_K2_DR32)
     F             /(K1*K1+K2*K2)
C
      D_CZ2S2_DR10=(D_K3_DR10*RS2+K3*D_RS2_DR10+D_K4_DR10*RC2+
     F                           K4*D_RC2_DR10)/(K4*K3+K4*K4)-
     F             (K3*RS2+K4*RC2)*(2.*K3*D_K3_DR10+2.*K4*D_K4_DR10)
     F             /(K3*K3+K4*K4)
      D_CZ2S2_DR20=(D_K3_DR20*RS2+K3*D_RS2_DR20+D_K4_DR20*RC2+
     F                           K4*D_RC2_DR20)/(K3*K3+K4*K4)-
     F             (K3*RS2+K4*RC2)*(2.*K3*D_K3_DR20+2.*K4*D_K4_DR20)
     F             /(K3*K3+K4*K4)
      D_CZ2S2_DR30=(D_K3_DR30*RS2+K3*D_RS2_DR30+D_K4_DR30*RC2+
     F                           K4*D_RC2_DR30)/(K3*K3+K4*K4)-
     F             (K3*RS2+K4*RC2)*(2.*K3*D_K3_DR30+2.*K4*D_K4_DR30)
     F             /(K3*K3+K4*K4)
      D_CZ2S2_DR12=(D_K3_DR12*RS2+K3*D_RS2_DR12+D_K4_DR12*RC2+
     F                           K4*D_RC2_DR12)/(K3*K3+K4*K4)-
     F             (K3*RS2+K4*RC2)*(2.*K3*D_K3_DR12+2.*K4*D_K4_DR12)
```

```
     F             /(K3*K3+K4*K4)
      D_CZ2S2_DR22=(D_K3_DR22*RS2+K3*D_RS2_DR22+D_K4_DR22*RC2+
     F                          K4*D_RC2_DR22)/(K3*K3+K4*K4)-
     F             (K3*RS2+K4*RC2)*(2.*K3*D_K3_DR22+2.*K4*D_K4_DR22)
     F             /(K3*K3+K4*K4)
      D_CZ2S2_DR32=(D_K3_DR32*RS2+K3*D_RS2_DR32+D_K4_DR32*RC2+
     F                          K4*D_RC2_DR32)/(K3*K3+K4*K4)-
     F             (K3*RS2+K4*RC2)*(2.*K3*D_K3_DR32+2.*K4*D_K4_DR32)
     F             /(K3*K3+K4*K4)
C
      D_CZ2C2_DR10=(D_K3_DR10*RC2+K3*D_RC2_DR10-D_K4_DR10*RS2-
     F                          K4*D_RS2_DR10)/(K3*K3+K4*K4)-
     F             (K3*RC2-K4*RS2)*(2.*K3*D_K3_DR10+2.*K4*D_K4_DR10)
     F             /(K3*K3+K4*K4)
      D_CZ2C2_DR20=(D_K3_DR20*RC2+K3*D_RC2_DR20-D_K4_DR20*RS2-
     F                          K4*D_RS2_DR20)/(K3*K3+K4*K4)-
     F             (K3*RC2-K4*RS2)*(2.*K3*D_K3_DR20+2.*K4*D_K4_DR20)
     F             /(K3*K3+K4*K4)
      D_CZ2C2_DR30=(D_K3_DR30*RC2+K3*D_RC2_DR30-D_K4_DR30*RS2-
     F                          K4*D_RS2_DR30)/(K3*K3+K4*K4)-
     F             (K3*RC2-K4*RS2)*(2.*K3*D_K3_DR30+2.*K4*D_K4_DR30)
     F             /(K3*K3+K4*K4)
      D_CZ2C2_DR12=(D_K3_DR12*RC2+K3*D_RC2_DR12-D_K4_DR12*RS2-
     F                          K4*D_RS2_DR12)/(K3*K3+K4*K4)-
     F             (K3*RC2-K4*RS2)*(2.*K3*D_K3_DR12+2.*K4*D_K4_DR12)
     F             /(K3*K3+K4*K4)
      D_CZ2C2_DR22=(D_K3_DR22*RC2+K3*D_RC2_DR22-D_K4_DR22*RS2-
     F                          K4*D_RS2_DR22)/(K3*K3+K4*K4)-
     F             (K3*RC2-K4*RS2)*(2.*K3*D_K3_DR22+2.*K4*D_K4_DR22)
     F             /(K3*K3+K4*K4)
      D_CZ2C2_DR32=(D_K3_DR32*RC2+K3*D_RC2_DR32-D_K4_DR32*RS2-
     F                          K4*D_RS2_DR32)/(K3*K3+K4*K4)-
     F             (K3*RC2-K4*RS2)*(2.*K3*D_K3_DR32+2.*K4*D_K4_DR32)
     F             /(K3*K3+K4*K4)
C
      D_CZ20_DR10=D_R0_DR10/R2 - R0/R2/R2*D_R2_DR10
      D_CZ20_DR20=D_R0_DR20/R2 - R0/R2/R2*D_R2_DR20
      D_CZ20_DR30=D_R0_DR30/R2 - R0/R2/R2*D_R2_DR30
      D_CZ20_DR12=D_R0_DR12/R2 - R0/R2/R2*D_R2_DR12
      D_CZ20_DR22=D_R0_DR22/R2 - R0/R2/R2*D_R2_DR22
      D_CZ20_DR32=D_R0_DR32/R2 - R0/R2/R2*D_R2_DR32
C
      D_P2_DR10=D_CZ20_DR10+D_CZ2S1_DR10*SIN(PHI)+
     F                      D_CZ2C1_DR10*COS(PHI)+
     F                      D_CZ2S2_DR10*SIN(2.*PHI)+
     F                      D_CZ2C2_DR10*COS(2.*PHI)
      D_P2_DR20=D_CZ20_DR20+D_CZ2S1_DR20*SIN(PHI)+
     F                      D_CZ2C1_DR20*COS(PHI)+
     F                      D_CZ2S2_DR20*SIN(2.*PHI)+
     F                      D_CZ2C2_DR20*COS(2.*PHI)
      D_P2_DR30=D_CZ20_DR30+D_CZ2S1_DR30*SIN(PHI)+
     F                      D_CZ2C1_DR30*COS(PHI)+
     F                      D_CZ2S2_DR30*SIN(2.*PHI)+
     F                      D_CZ2C2_DR30*COS(2.*PHI)
      D_P2_DR12=D_CZ20_DR12+D_CZ2S1_DR12*SIN(PHI)+
     F                      D_CZ2C1_DR12*COS(PHI)+
     F                      D_CZ2S2_DR12*SIN(2.*PHI)+
     F                      D_CZ2C2_DR12*COS(2.*PHI)
      D_P2_DR22=D_CZ20_DR22+D_CZ2S1_DR22*SIN(PHI)+
     F                      D_CZ2C1_DR22*COS(PHI)+
     F                      D_CZ2S2_DR22*SIN(2.*PHI)+
     F                      D_CZ2C2_DR22*COS(2.*PHI)
      D_P2_DR32=D_CZ20_DR32+D_CZ2S1_DR32*SIN(PHI)+
     F                      D_CZ2C1_DR32*COS(PHI)+
     F                      D_CZ2S2_DR32*SIN(2.*PHI)+
     F                      D_CZ2C2_DR32*COS(2.*PHI)
C
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE MRQMIN(X,Y,SIG,NDATA,A,IA,MA,COVAR,ALPHA,NCA,CHISQ,
     *FLIFT,ALAMDA)
C     LEVENBERG-MARQUARDT METHOD, ATTEMPTING TO REDUCE THE VALUE CHISQ
C     OF A FIT BETWEEN A SET OF DATA POINTS X(1:NDATA),Y(1:NDATA) WITH
C     INDIVIDUAL STANDARD DEVIATIONS SIG(1:NDATA), AND A NONLINEAR FUNCTION
C     DEPENDENT ON MA COEFFICIENTS A(1:MA). THE INPUT ARRAY IA(1:MA)
C     INDICATES BY NON-ZERO ENTRIES THOSE COMPONENTS THAT SHOULD BE HELD
C     FIXED AT THEIR INPUT VALUES. THE PROGRAM RETURNS CURRENT BEST-FIT
```

```
C     VALUES FOR THE PARAMETERS A(1:MA), AND CHISQ. THE ARRAYS COVAR(1:NCA,
C     1:NCA), ALPHA(1:NCA,1:NCA) WITH PHYSICAL DIMENSION NCA(>= THE NUMBER
C     OF FITTED PARAMETERS) ARE USED AS WORKING SPACE DURING MOST ITERATIONS.
C     SUPPLY A SUBROUTINE FLIFT(X,XPREV,A,NF,Y,DYDA,NA)THAT EVALUATES THE FITTING FUNCTION
C     YFIT, AND ITS DERIVATIVES DYDA WITH RESPECT TO THE FITTING PARAMETERS
C     A AT X. ON THE FIRST CALL PROVIDE AN INITIAL GUESS FOR THE PARAMETERS
C     A, AND SET ALAMDA<0 FOR INITIALIZATION (WHICH THEN SETS ALAMDA=0.001).
C     IF A STEP SUCCEEDS CHISQ BECOMES SMALLER AND ALAMDA DECREASES BY A
C     FACTOR OF 10. IF A STEP FAILS ALAMDA GROWS BY A FACTOR OF 10. YOU MUST
C     CALL THIS ROUTINE REPEATEDLY UNTIL CONVERGENCE IS ACHIEVED. THEN,
C     MAKE ONE FINAL CALL WITH ALAMDA=0, SO THAT COVAR(1:MA,1:MA) RETURNS
C     THE COVARIANCE MATRIX, AND ALPHA THE CURVATURE MATRIX. (PARAMETERS
C     HELD FIXED WILL RETURN ZERO COVARIANCES)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      INTEGER MA,NCA,NDATA,IA(MA),MMAX
      REAL ALAMDA,CHISQ,A(MA),ALPHA(NCA,NCA),COVAR(NCA,NCA),
     *SIG(NDATA),X(NDATA),Y(NDATA)
      EXTERNAL FLIFT
      PARAMETER (MMAX=20)
      INTEGER J,K,L,MFIT
      REAL OCHISQ,ATRY(MMAX),BETA(MMAX),DA(MMAX)
      SAVE OCHISQ,ATRY,BETA,DA,MFIT
      IF(ALAMDA.LT.0.)THEN
        MFIT=0
        DO J=1,MA
          IF (IA(J).NE.0) MFIT=MFIT+1
        ENDDO
        ALAMDA=0.001
        CALL MRQCOF(X,Y,SIG,NDATA,A,IA,MA,ALPHA,BETA,NCA,CHISQ,FLIFT)
        OCHISQ=CHISQ
        DO 12 J=1,MA
          ATRY(J)=A(J)
12       CONTINUE
      ENDIF
      DO 14 J=1,MFIT
          DO 13 K=1,MFIT
              COVAR(J,K)=ALPHA(J,K)
13         CONTINUE
          COVAR(J,J)=ALPHA(J,J)*(1.+ALAMDA)
          DA(J)=BETA(J)
14    CONTINUE
      CALL GAUSSJ(COVAR,MFIT,NCA,DA,1,1)
      IF(ALAMDA.EQ.0.)THEN
        CALL COVSRT(COVAR,NCA,MA,IA,MFIT)
        RETURN
      ENDIF
      J=0
      DO 15 L=1,MA
        IF(IA(L).NE.0) THEN
          J=J+1
          ATRY(L)=A(L)+DA(J)
        ENDIF
15    CONTINUE
      CALL MRQCOF(X,Y,SIG,NDATA,ATRY,IA,MA,COVAR,DA,NCA,CHISQ,FLIFT)
      PRINT*,OCHISQ,CHISQ
      IF(CHISQ.LT.OCHISQ)THEN
        ALAMDA=0.1*ALAMDA
        OCHISQ=CHISQ
        DO 17 J=1,MFIT
            DO 16 K=1,MFIT
               ALPHA(J,K)=COVAR(J,K)
16          CONTINUE
            BETA(J)=DA(J)
17       CONTINUE
        DO 18 L=1,MA
            A(L)=ATRY(L)
18       CONTINUE
      ELSE
        ALAMDA=10.*ALAMDA
        CHISQ=OCHISQ
      ENDIF
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE MRQCOF(X,Y,SIG,NDATA,A,IA,MA,ALPHA,BETA,NALP,CHISQ,
     *FLIFT)
C     USED BY MRQMIN TO EVALUATE THE LINEARIZED FITTING MATRIX ALPHA
C     AND VECTOR BETA AND CALCULATE CHISQ.
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      INTEGER MA,NALP,NDATA,IA(MA),MMAX
      REAL CHISQ,A(MA),ALPHA(NALP,NALP),BETA(MA),SIG(NDATA),X(NDATA),
     *Y(NDATA)
      EXTERNAL FLIFT
      PARAMETER (MMAX=20)
      INTEGER MFIT,I,J,K,L,M
      REAL DY,SIG2I,WT,YMOD,DYDA(MMAX)
      MFIT=0
      DO 11 J=1,MA
        IF (IA(J).NE.0) MFIT=MFIT+1
11    CONTINUE
      DO 13 J=1,MFIT
        DO 12 K=1,J
          ALPHA(J,K)=0.
12      CONTINUE
        BETA(J)=0.
13    CONTINUE
      CHISQ=0.
      DO 16 I=1,NDATA
        NF=0
        IF (I.EQ.1) THEN
CGSOS           CALL FLIFT(X(I),-13.,A,NF,YMOD,DYDA,MA)
          CALL FLIFT(X(I),X(NDATA),A,NF,YMOD,DYDA,MA)
        ELSE
          CALL FLIFT(X(I),X(I-1),A,NF,YMOD,DYDA,MA)
        ENDIF
        SIG2I=1./(SIG(I)*SIG(I))
        DY=Y(I)-YMOD
        J=0
        DO 15 L=1,MA
          IF(IA(L).NE.0) THEN
            J=J+1
            WT=DYDA(L)*SIG2I
            K=0
            DO 14 M=1,L
              IF (IA(M).NE.0) THEN
                K=K+1
                ALPHA(J,K)=ALPHA(J,K)+WT*DYDA(M)
              ENDIF
14          CONTINUE
            BETA(J)=BETA(J)+DY*WT
          ENDIF
15      CONTINUE
        CHISQ=CHISQ+DY*DY*SIG2I
16    CONTINUE
      DO 18 J=2,MFIT
        DO 17 K=1,J-1
          ALPHA(K,J)=ALPHA(J,K)
17      CONTINUE
18    CONTINUE
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE FLIFT(X,XPREV,A,NF,Y,DYDA,NA)
C     CALCULATES THE LIFT COEFFICIENT (Y) FOR EVERY ANGLE OF ATTACK
C     (X) AND THE CURRENT A (ACTUALLY Ri) PARAMETERS. IT ALSO CALCULATES
C     THE DERIVATIVES DYDA.
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      INTEGER NA
      REAL X,Y,A(NA),DYDA(NA)
      REAL PHI,K,THETA,F,G,CLGO,CLS,CLC,CL1,CL2,
     F     ALPHA0D,ALPHAVD,ALPHA0,AVIB,R1,R2,R3
      COMMON/FUN/K,ALPHA0D,ALPHAVD
C
      ALPHA0=ALPHA0D*3.141593/180.0
      AVIB  =ALPHAVD*3.141593/180.0
      PHI=ASIN((X-ALPHA0D)/ALPHAVD)
CGSOS      IF (XPREV.EQ.-13.) GOTO 5
      PREVPHI=ASIN((XPREV-ALPHA0D)/ALPHAVD)
      IF (PHI.GT.PREVPHI) PHI=3.141593-PHI
5     CALL FUNCVL(ALPHA0,AVIB,K,PHI,THETA,CLGO,CLS,CLC,F,G,CL1)
      CALL FUNCVN(A,NA,ALPHA0,AVIB,R1,R2,R3,K,PHI,CZ2C2,
     F            CZ2S2,CZ2C1,CZ2S1,CZ20,CL2,
     F D_P2_DR10,D_P2_DR20,D_P2_DR30,D_P2_DR12,D_P2_DR22,D_P2_DR32)
      Y=CL1+CL2
      IF (NF.EQ.1) WRITE(9,4000)X,Y,CL1,CL2
      DYDA(1)=D_P2_DR10
```

```
      DYDA(2)=D_P2_DR20
      DYDA(3)=D_P2_DR30
      DYDA(4)=D_P2_DR12
      DYDA(5)=D_P2_DR22
      DYDA(6)=D_P2_DR32
C
4000  FORMAT(1X,4(1X,F10.4))
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE COVSRT(COVAR,NPC,MA,IA,MFIT)
C     THIS IS MERELY FOR REARRANGING THE COVARIANCE MATRIX COVAR INTO
C     THE ORDER OF ALL MA PARAMETERS.
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      INTEGER MA,MFIT,NPC,IA(MA)
      REAL COVAR(NPC,NPC)
      INTEGER I,J,K
      REAL SWAP
      DO 12 I=MFIT+1,MA
        DO 11 J=1,I
          COVAR(I,J)=0.
          COVAR(J,I)=0.
11       CONTINUE
12    CONTINUE
      K=MFIT
      DO 15 J=MA,1,-1
        IF(IA(J).NE.0)THEN
          DO 13 I=1,MA
            SWAP=COVAR(I,K)
            COVAR(I,K)=COVAR(I,J)
            COVAR(I,J)=SWAP
13         CONTINUE
          DO 14 I=1,MA
            SWAP=COVAR(K,I)
            COVAR(K,I)=COVAR(J,I)
            COVAR(J,I)=SWAP
14         CONTINUE
          K=K-1
        ENDIF
15    CONTINUE
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE GAUSSJ(A,N,NP,B,M,MP)
C     LINEAR EQUATION SOLUTION BY GAUSS-JORDAN ELIMINATION. THE INPUT
C     MATRIX A(1:N,1:N) HAS N BY N ELEMENTS. B(1:N,1:M) IS AN INPUT MATRIX
C     OF SIZE N BY M CONTAINING THE M RIGHT-HAND SIDE VECTORS. ON OUTPUT,
C     A IS REPLACED BY ITS MATRIX INVERSE, AND B IS REPLACED BY THE
C     CORRESPONDING SET OF SOLUTION VECTORS.
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      INTEGER M,MP,N,NP,NMAX
      REAL A(NP,NP),B(NP,MP)
      PARAMETER (NMAX=500)
      INTEGER I,ICOL,IROW,J,K,L,LL,INDXC(NMAX),INDXR(NMAX),IPIV(NMAX)
      REAL BIG,DUM,PIVINV
      DO 11 J=1,N
         IPIV(J)=0
11    CONTINUE
      DO 22 I=1,N
         BIG=0.
         DO 13 J=1,N
           IF (IPIV(J).NE.1)THEN
              DO 12 K=1,N
                 IF (IPIV(K).EQ.0) THEN
                    IF (ABS(A(J,K)).GE.BIG)THEN
                       BIG=ABS(A(J,K))
                       IROW=J
                       ICOL=K
                    ENDIF
                 ELSE IF (IPIV(K).GT.1) THEN
                    PAUSE '1 SINGULAR MATRIX IN GAUSSJ'
                 ENDIF
12            CONTINUE
           ENDIF
13       CONTINUE
        IPIV(ICOL)=IPIV(ICOL)+1
        IF (IROW.NE.ICOL) THEN
          DO 14 L=1,N
```

```
            DUM=A(IROW,L)
            A(IROW,L)=A(ICOL,L)
            A(ICOL,L)=DUM
14          CONTINUE
           DO 15 L=1,M
            DUM=B(IROW,L)
            B(IROW,L)=B(ICOL,L)
            B(ICOL,L)=DUM
15         CONTINUE
          ENDIF
          INDXR(I)=IROW
          INDXC(I)=ICOL
          IF (A(ICOL,ICOL).EQ.0.) PAUSE '0 SINGULAR MATRIX IN GAUSSJ'
          PIVINV=1./A(ICOL,ICOL)
          A(ICOL,ICOL)=1.
          DO 16 L=1,N
            A(ICOL,L)=A(ICOL,L)*PIVINV
16         CONTINUE
          DO 17 L=1,M
            B(ICOL,L)=B(ICOL,L)*PIVINV
17         CONTINUE
          DO 21 LL=1,N
            IF(LL.NE.ICOL)THEN
              DUM=A(LL,ICOL)
              A(LL,ICOL)=0.
              DO 18 L=1,N
                A(LL,L)=A(LL,L)-A(ICOL,L)*DUM
18             CONTINUE
              DO 19 L=1,M
                B(LL,L)=B(LL,L)-B(ICOL,L)*DUM
19             CONTINUE
            ENDIF
21        CONTINUE
22      CONTINUE
        DO 24 L=N,1,-1
          IF(INDXR(L).NE.INDXC(L))THEN
            DO 23 K=1,N
              DUM=A(K,INDXR(L))
              A(K,INDXR(L))=A(K,INDXC(L))
              A(K,INDXC(L))=DUM
23           CONTINUE
          ENDIF
24      CONTINUE
        RETURN
        END
```

4.2. Flutter program

```
      PROGRAM FLUTTER
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C     3-D Flutter Calculation
C
C     Input:
C
C     COEFFICIENTS OF THE ONERA NON-LINEAR AERODYNAMIC EQUATIONS
C
C     R10  -5.6958   R10 ONERA NON-LINEAR COEFFICIENT
C     R20   0.6425   R20 ONERA NON-LINEAR COEFFICIENT
C     R30  -0.8193   R30 ONERA NON-LINEAR COEFFICIENT
C     R12 -17.4499   R12 ONERA NON-LINEAR COEFFICIENT
C     R22   1.6833   R22 ONERA NON-LINEAR COEFFICIENT
C     R32  16.3939   R32 ONERA NON-LINEAR COEFFICIENT
C
C     AERODYNAMIC INTEGRALS
C
C     I1 1.0         I1 AERODYNAMIC INTEGRAL
C     I2 0.6779      I2 AERODYNAMIC INTEGRAL
C     I3 0.5         I3 AERODYNAMIC INTEGRAL
C     I4 0.783       I4 AERODYNAMIC INTEGRAL
C     I5 0.6366      I5 AERODYNAMIC INTEGRAL
C
C     LINEAR AERODYNAMIC COEFFICIENTS LIFT/MOMENT
C
C     LAM1 0.15      LAM1 ONERA LINEAR COEFFICIENT FOR LIFT/MOMENT
C     LAM2 0.55      LAM2 ONERA LINEAR COEFFICIENT FOR LIFT/MOMENT
C     SL1  3.1415    SL1  ONERA LINEAR COEFFICIENT FOR LIFT
C     SL2  1.571     SL2  ONERA LINEAR COEFFICIENT FOR LIFT
C     SL3  0.0       SL3  ONERA LINEAR COEFFICIENT FOR LIFT
C     AOL  5.73      AOL  ONERA LINEAR COEFFICIENT FOR LIFT
C     SM1 -0.786     SM1 ONERA LINEAR COEFFICIENT FOR MOMENT
C     SM2 -0.589     SM2 ONERA LINEAR COEFFICIENT FOR MOMENT
C     SM3 -0.786     SM3 ONERA LINEAR COEFFICIENT FOR MOMENT
C     AOM 0.0        AOM ONERA LINEAR COEFFICIENT FOR MOMENT
C     B1 8.6         NON-LINEAR SLOPE OF THE DEVIATION FROM LINEAR FORCE CURVE  - LIFT
C     B2 0.4         NON-LINEAR SLOPE OF THE DEVIATION FROM LINEAR FORCE CURVE  - MOMENT
C
C     STRUCTURE - MATERIALS
C
C     M   0.283      TOTAL MASS PER UNIT LENGTH (kg/m)
C     IA  0.343E-3   MASS MOMENT OF INERTIA PER UNIT LENGTH (kg m)
C     WH  27.02      UNCOUPLED BENDING FREQUENCY (rad/s)
C     WA  154.6      UNCOUPLED TORSIONAL FREQUENCY  (rad/s)
C     RHO 1.23       WING MATERIAL DENSITY FREE STREAM DENSITY (kg/m^3)
C
C     GEOMETRY
C
C     C   0.140      CHORD LENGTH (m)
C     L   0.559      WING SPAN (m)
C     ALPHA1D  10.   STALL ANGLE DEGREES
C     THETA0D  0.    ROOT ANGLE DEGREES
C
C     INITIAL GUESS
C
C     x(1)=Q10       MEAN COMPONENT FIRST MODE (BENDING)
C     x(2)=Q20       MEAN COMPONENT SECOND MODE (TORSION)
C     x(3)=Q1S       SIN  COMPONENT FIRST MODE
C     x(4)=Q1C       COS  COMPONENT FIRST MODE
C     x(5)=K2        TORSIONAL REDUCED FREQUENCY
C     x(6)=K         REDUCED FREQUENCY
C
C     MODE SHAPE
C
C     PHIA 0.84460   BEAM TORSION MODE SHAPE
C     PHIH 1.31538   BEAM BENDING MODE SHAPE
C
C     CONVERGENCE
C
C     NITS  25       MAX NUMBER OF NEWTON-RAPHSON ITERATIONS
```

```
C     TOLER 1.E-4    TOLERANCE FOR EACH OF THE 6 UNKNOWNS
C
C     PARAMETERIC STUDY
C
C     Q2C 0.0        COS COMPONENT OF THE SECOND MODAL FORCE
C     Q2S 0.1        SINE COMPONENT OF THE SECOND MODAL FORCE
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      PARAMETER (NN=6)
      INTEGER N,NP,I,T,J,INDX(NN),NITS
      REAL*8
     1      X(1:NN),FVEC(1:NN),FJAC(1:NN,1:NN),DELTA(NN),FS,FQ,AVIB,
     2      ALPHA0,PHI1T,PHI1,TOLER
      REAL  Y(NN,NN),D,THETA0,THETA0D,Q2S
C
      REAL
     1      R10,R20,R30,R12,R22,R32,
     2      I1,I2,I3,I4,I5,
     3      LAM1,LAM2,SL1,SL2,SL3,AOL,SM1,SM2,SM3,AOM,B1,B2,
     4      M,IA,RHO,C,L,WH,WA,ALPHA1D,ALPHA1,ALPHA1T,B,PI,OM,MU,RA,
     5      PHIA,PHIH,Q2C
      COMMON/USER/
     1      R10,R20,R30,R12,R22,R32,
     2      I1,I2,I3,I4,I5,
     3      LAM1,LAM2,SL1,SL2,SL3,AOL,SM1,SM2,SM3,AOM,B1,B2,
     4      M,IA,RHO,C,L,WH,WA,ALPHA1D,ALPHA1,ALPHA1T,B,PI,OM,MU,RA,
     5      PHIA,PHIH,Q2C
C
C     READ DATA FROM THE INPUT FILE
C
      OPEN (10,FILE='aeroel3.in',STATUS='OLD')
      READ (10,120) R10,R20,R30,R12,R22,R32
      READ (10,120) I1,I2,I3,I4,I5
      READ (10,120) LAM1,LAM2,SL1,SL2,SL3,AOL,SM1,SM2,SM3,AOM,B1,B2
      READ (10,120) M,IA,RHO,C,L,WH,WA,ALPHA1D,THETA0D
      READ (10,120) X(1),X(2),X(3),X(4),X(5),X(6)
      READ (10,120) PHIA,PHIH
      READ (10,*)   NITS
      READ (10,120) TOLER
      READ (10,120) Q2C,Q2S
C
C     CREATE OUTPUT FILE
C
      OPEN  (9,FILE='aeroel3.out',STATUS='UNKNOWN')
      WRITE (9,*)
      WRITE (9,*) '# 3D Flutter Calculation'
      WRITE (9,*)
      WRITE (9,105)
     1      'Q2S','Q10','Q20','Q1S','Q1C','K2','K',
     2      'FS (M/S)','W (Hz)','A0','AV'
C
C SETUP PARAMETERS
C
      THETA0=DEG2RAD(THETA0D)
      N=6
      NP=6
C
      ALPHA1=DEG2RAD(ALPHA1D)
      ALPHA1T=-ALPHA1
      B=0.5*C
      PI=3.14159
      OM=WH/WA
      MU=(M/L)/(PI*RHO*B*B)
      RA=SQRT(IA/M)/B
C
C     NEWTON STEP
C
      T=0
      DO I=1,N
         DELTA(I)=5.
      ENDDO
20    CONTINUE
      print *, 'START',T
      IF (DABS(DELTA(1)).LE.TOLER.AND.
     1    DABS(DELTA(2)).LE.TOLER.AND.
     2    DABS(DELTA(3)).LE.TOLER.AND.
     3    DABS(DELTA(4)).LE.TOLER.AND.
     4    DABS(DELTA(5)).LE.TOLER.AND.
```

```
     5    DABS(DELTA(6)).LE.TOLER) GOTO 90
      T=T+1
C
C     CALCULATE THE FUNCTIONS USING THE INITIAL VALUES FOR THE
C     STATE VECTOR X
C
      CALL FUNCV(THETA0,Q2S,ALPHA0,AVIB,PHI1,PHI1T,N,X,FVEC)
C
C     CALCULATE THE JACOBIAN MATRIX
C
      CALL FDJAC(Q2S,THETA0,N,X,FVEC,NP,FJAC)
C
      DO I=1,N
         DO J=1,N
            Y(I,J)=0.
         ENDDO
         Y(I,I)=1.
      ENDDO
C
C     CALCULATE THE INVERSE OF THE JACOBIAN MATRIX
C
      CALL LUDCMP (FJAC,N,NP,INDX,D)
      DO J=1,N
         CALL LUBKSB (FJAC,N,NP,INDX,Y(1,J))
      ENDDO
C
C     CALCULATE DELTA
C
      DO J=1,N
         DELTA(J)=0.
         DO I=1,N
            DELTA(J)=DELTA(J)+Y(J,I)*FVEC(I)
         ENDDO
      ENDDO
C
C     CALCULATE X AT THE NEXT ITERATION
C
      DO I=1,N
         X(I)=X(I)-DELTA(I)
      ENDDO
C
      IF (T.LT.NITS) GOTO 20
C
C     PRINTOUT
C
      CONTINUE
      print *, T
      IF (T.GT.NITS) THEN
         WRITE (*,*) 'DIVERGENCE'
         STOP
      ENDIF
C
C     CALCULATE FLUTTER VELOCITY AND FREQUENCY
C
90    FS=WA*B/X(5)
      FQ=FS*X(6)/(B*2.*PI)
      WRITE (9,110)
     1      Q2S,X(1),X(2),X(3),X(4),X(5),X(6),
     2      FS,FQ,RAD2DEG(ALPHA0),RAD2DEG(AVIB)
C
C     NEXT Q2S ENTRY
C
      DO I=1,N
         X(I)=X(I)+0.0001
      ENDDO
      READ (10,*) Q2S
      T=0
      DO I=1,N
         DELTA(I)=5.
      ENDDO
      IF (Q2S.LT.999.0) GOTO 20
C
      STOP
C
C     FORMAT STATEMENTS
C
105   FORMAT('#',11(1X,A11))
110   FORMAT(11(1X,F11.4))
```

```
120   FORMAT(1X,F10.5)
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE FDJAC (Q2S,THETA0,N,X,FVEC,NP,DF)
C
C     THE JACOBIAN MATRIX IS CALCULATED INCREASING THE STATE VECTOR
C     BY A SMALL NUMBER
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL*8 DF(NP,NP),FVEC(N),X(N),EPS,F(40),ALPHA0,AVIB,PHI1,PHI1T
      REAL H,TEMP,THETA0
      EPS=1.E-4
      DO J=1,N
         TEMP=X(J)
         H=EPS*ABS(TEMP)
         IF (H.EQ.0.) H=EPS
         X(J)=TEMP+H
         H=X(J)-TEMP
         CALL FUNCV (THETA0,Q2S,ALPHA0,AVIB,PHI1,PHI1T,N,X,F)
         X(J)=TEMP
         DO I=1,N
            DF(I,J)=(F(I)-FVEC(I))/H
         ENDDO
      ENDDO
C
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE FUNCV (THETA0,Q2S,ALPHA0,AVIB,PHI1,PHI1T,N,X,P)
C
C     THE AEROELASTIC MODEL IS USED HERE TO FORMULATE THE EQUATIONS
C     WHICH ARE TO BE SOLVED (P). THE VARIABLES CAN BE EASILY IDENTIFIED
C     WITH THE NAME OF THE VARIABLES IN THE DISSERTATION.
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      INTEGER N
      REAL Q2S,THETA0
      REAL*8 X(N),P(N),CL20,CM20,ALPHA0,ALPHAS,ALPHAC,AVIB,F,G,
     1BOL,B1L,B2L,B3L,B4L,BOM,B1M,B2M,B3M,B4M,PENT,PENT2,PHI1,PHI1T,
     2DCL0,DCLS1,R1L,R2L,R3L,K1L,K2L,RS1L,RC1L,ALS1,BLC1,DCM0,DCMS1,R1M,
     3R2M,R3M,K1M,K2M,RS1M,RC1M,AMS1,BMC1
C
      REAL
     1      R10,R20,R30,R12,R22,R32,
     2      I1,I2,I3,I4,I5,
     3      LAM1,LAM2,SL1,SL2,SL3,AOL,SM1,SM2,SM3,AOM,B1,B2,
     4      M,IA,RHO,C,L,WH,WA,ALPHA1D,ALPHA1,ALPHA1T,B,PI,OM,MU,RA,
     5      PHIA,PHIH,Q2C
      COMMON/USER/
     1      R10,R20,R30,R12,R22,R32,
     2      I1,I2,I3,I4,I5,
     3      LAM1,LAM2,SL1,SL2,SL3,AOL,SM1,SM2,SM3,AOM,B1,B2,
     4      M,IA,RHO,C,L,WH,WA,ALPHA1D,ALPHA1,ALPHA1T,B,PI,OM,MU,RA,
     5      PHIA,PHIH,Q2C
C
C     ANGLE OF ATTACK IN STRUCTURAL CO-ORDINATES
C
      ALPHA0=THETA0+.5*PHIA*X(2)
      ALPHAS=PHIA*(.5*Q2S+.25*X(6)*Q2C)+PHIH*X(6)*X(4)
      ALPHAC=PHIA*(.5*Q2C-.25*X(6)*Q2S)-PHIH*X(6)*X(3)
      AVIB=DSQRT(ALPHAS*ALPHAS+ALPHAC*ALPHAC)
C
C     LINEAR PART - L REFERS TO LIFT
C
      F=(X(6)*X(6)*LAM2+LAM1*LAM1)/(LAM1*LAM1+X(6)*X(6))
      G=(X(6)*LAM1*(LAM2-1))/(LAM1*LAM1+X(6)*X(6))
      BOL=X(2)*AOL*.5
      B1L=X(6)*(.25*X(6)*SL1-.5*X(6)*SL2-.25*G*AOL)+.5*F*AOL
      B2L=X(6)*(.5*SL1+.5*SL3+.25*F*AOL)+.5*G*AOL
      B3L=X(6)*(SL1*X(6)+G*AOL)
      B4L=F*AOL*X(6)
C
C     LINEAR PART - M REFERS TO MOMENT
C
      BOM=X(2)*AOM*.5
      B1M=X(6)*(.25*X(6)*SM1-.5*X(6)*SM2-.25*G*AOM)+.5*F*AOM
      B2M=X(6)*(0.5*SM1+0.5*SM3+0.25*F*AOM)+.5*G*AOM
      B3M=X(6)*(SM1*X(6)+G*AOM)
      B4M=F*AOM*X(6)
C
```

```
C      SINGLE BREAK-POINT
C
       PENT=(ALPHA1-ALPHA0)/AVIB
       IF (PENT.GT.1.) THEN
          PHI1=0.5*PI
       ELSE IF (PENT.LT.-1.) THEN
          PHI1=-0.5*PI
       ELSE
          PHI1=DASIN(PENT)
       ENDIF
C
       PENT2=(ALPHA1T-ALPHA0)/AVIB
       IF (PENT2.GT.1.) THEN
          PHI1T=0.5*PI
       ELSE IF (PENT2.LT.-1.) THEN
          PHI1T=-0.5*PI
       ELSE
          PHI1T=DASIN(PENT2)
       ENDIF
C
C      NON-LINEAR PART - L REFERS TO LIFT
C
       DCL0 =(B1*AVIB/PI)*(-PENT*(1.571-PHI1 )+DCOS(PHI1 ))
      F      -(B1*AVIB/PI)*(PENT2*(1.571+PHI1T)+DCOS(PHI1T))
       DCLS1=(B1*AVIB/PI)*( (1.571-PHI1 )-.5*DSIN(2.*PHI1 ))
      F      -(B1*AVIB/PI)*(-(1.571+PHI1T)-.5*DSIN(2.*PHI1T))
       CL20=-DCL0
       R1L=R10+R12*DCL0*DCL0
       R2L=(R20+R22*DCL0*DCL0)**2
       R3L=(R30+R32*DCL0*DCL0)*R2L
       K1L=R2L-X(6)*X(6)
       K2L=R1L*X(6)
       RS1L=-R2L*DCLS1
       RC1L=-R3L*X(6)*DCLS1
       ALS1=(K1L*RS1L+K2L*RC1L)/(K1L*K1L+K2L*K2L)
       BLC1=(K1L*RC1L-K2L*RS1L)/(K1L*K1L+K2L*K2L)
C
C      NON-LINEAR PART - M REFERS TO MOMENT
C
       DCM0= (B2*AVIB/PI)*(-PENT*(1.571-PHI1 )+DCOS(PHI1 ))-
      F       (B2*AVIB/PI)*(PENT2*(1.571+PHI1T)+DCOS(PHI1T))
       DCMS1=(B2*AVIB/PI)*( (1.571-PHI1 )-0.5*DSIN(2.*PHI1 ))-
      F       (B2*AVIB/PI)*(-(1.571+PHI1T)-0.5*DSIN(2.*PHI1T))
       CM20=-DCM0
       R1M=R10+R12*DCM0*DCM0
       R2M=(R20+R22*DCM0*DCM0)**2
       R3M=(R30+R32*DCM0*DCM0)*R2M
       K1M=R2M-X(6)*X(6)
       K2M=R1M*X(6)
       RS1M=-R2M*DCMS1
       RC1M=-R3M*X(6)*DCMS1
       AMS1=(K1M*RS1M+K2M*RC1M)/(K1M*K1M+K2M*K2M)
       BMC1=(K1M*RC1M-K2M*RS1M)/(K1M*K1M+K2M*K2M)
C
C      EQUATIONS TO SOLVE
C
       P(1)=MU*PI*I1*OM*OM*X(5)*X(5)*X(1)-I4*AOL*THETA0-I2*BOL-I4*CL20
C
       P(2)=MU*(PI/4.)*RA*RA*I3*X(5)*X(5)*X(2)-.25*I5*AOL*THETA0-.25*I3*
      1BOL-.25*I5*CL20-I5*CM20
C
       P(3)=MU*PI*I1*(-X(6)*X(6)*X(3)+OM*OM*X(5)*X(5)*X(3))-I2*(Q2S*B1L-
      1Q2C*B2L)-I1*(X(3)*B3L+X(4)*B4L)-I4*ALS1*ALPHAS/AVIB+I4*BLC1*
      2ALPHAC/AVIB
C
       P(4)=MU*PI*I1*(-X(6)*X(6)*X(4)+OM*OM*X(5)*X(5)*X(4))-I2*(Q2S*B2L+
      1Q2C*B1L)-I1*(-X(3)*B4L+X(4)*B3L)-I4*ALS1*ALPHAC/AVIB-I4*BLC1*
      2ALPHAS/AVIB
C
       P(5)=MU*(PI/4.)*RA*RA*I3*(-X(6)*X(6)*Q2S+X(5)*X(5)*Q2S)-.25*I3*
      1(Q2S*B1L-Q2C*B2L)-.25*I2*(X(3)*B3L+X(4)*B4L)-I3*(Q2S*B1M-Q2C*B2M)-
      2I2*(X(3)*B3M+X(4)*B4M)-I5*AMS1*ALPHAS/AVIB+I5*BMC1*ALPHAC/AVIB-
      3.25*I5*ALS1*ALPHAS/AVIB+.25*I5*BLC1*ALPHAC/AVIB
C
       P(6)=MU*(PI/4.)*RA*RA*I3*(-X(6)*X(6)*Q2C+X(5)*X(5)*Q2C)-.25*I3*
      1(Q2S*B2L+Q2C*B1L)-.25*I2*(-X(3)*B4L+X(4)*B3L)-I3*(Q2S*B2M+Q2C*B1M)
      2-I2*(-X(3)*B4M+X(4)*B3M)-I5*AMS1*ALPHAC/AVIB-I5*BMC1*ALPHAS/AVIB-
      3.25*I5*ALS1*ALPHAC/AVIB-.25*I5*BLC1*ALPHAS/AVIB
```

```
C
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE LUDCMP (A,N,NP,INDX,D)
C
C     LU MATRIX DECOMPOSITION
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      PARAMETER (NMAX=100,TINY=1.E-20)
      DOUBLE PRECISION A(NP,NP),VV(NMAX)
      DIMENSION INDX(N)
C
      D=1.
      DO I=1,N
         AAMAX=0.
         DO J=1,N
            IF (DABS(A(I,J)).GT.AAMAX) AAMAX=DABS(A(I,J))
         ENDDO
         IF (AAMAX.EQ.0.) PAUSE 'SINGULAR MATRIX IN LUDCMP'
         VV(I)=1./AAMAX
      ENDDO
      DO J=1,N
         DO I=1,J-1
            SUM=A(I,J)
            DO K=1,I-1
               SUM=SUM-A(I,K)*A(K,J)
            ENDDO
            A(I,J)=SUM
         ENDDO
         AAMAX=0.
         DO I=J,N
            SUM=A(I,J)
            DO K=1,J-1
               SUM=SUM-A(I,K)*A(K,J)
            ENDDO
            A(I,J)=SUM
            DUM=VV(I)*ABS(SUM)
            IF (DUM.GE.AAMAX) THEN
               IMAX=I
               AAMAX=DUM
            ENDIF
         ENDDO
         IF (J.NE.IMAX) THEN
            DO K=1,N
               DUM=A(IMAX,K)
               A(IMAX,K)=A(J,K)
               A(J,K)=DUM
            ENDDO
            D=-D
            VV(IMAX)=VV(J)
         ENDIF
         INDX(J)=IMAX
         IF (A(J,J).EQ.0.) A(J,J)=TINY
         IF (J.NE.N) THEN
            DUM=1./A(J,J)
            DO I=J+1,N
               A(I,J)=A(I,J)*DUM
            ENDDO
         ENDIF
      ENDDO
C
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE LUBKSB (A,N,NP,INDX,B)
C
C     BACK SUBSTITUTION
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      DIMENSION INDX(N),B(N)
      DOUBLE PRECISION A(NP,NP)
C
      II=0
      DO I=1,N
         LL=INDX(I)
         SUM=B(LL)
         B(LL)=B(I)
         IF (II.NE.0) THEN
            DO J=II,I-1
```

```
                  SUM=SUM-A(I,J)*B(J)
               ENDDO
            ELSE IF (SUM.NE.0.) THEN
               II=I
            ENDIF
            B(I)=SUM
         ENDDO
         DO I=N,1,-1
            SUM=B(I)
            DO J=I+1,N
               SUM=SUM-A(I,J)*B(J)
            ENDDO
            B(I)=SUM/A(I,I)
         ENDDO
C
         RETURN
         END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
         REAL FUNCTION DEG2RAD(ANGLE)
C
C        CONVERT DEG-->RAD
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
         REAL ANGLE
C
         DEG2RAD=ANGLE*ATAN(1.)/45.
C
         RETURN
         END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
         REAL FUNCTION RAD2DEG(ANGLE)
C
C        CONVERT RAD-->DEG
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
         REAL ANGLE
C
         RAD2DEG=ANGLE*45./ATAN(1.)
C
         RETURN
         END
```

4.3 Fourier components

```
      PROGRAM FOURIER
C
C     Fourier Coefficients for 1st and 2nd
C     Harmonics of Hysteresis Loops
C
CG      parameter(n=16,m=8)
      parameter(n=256,m=128)
CG      parameter(n=512,m=256)
      integer i
      real*8 f(1:n),c0,c1,s1,c2,s2
      open(5,file='aeroel5.in',status='old')
      open(9,file='aeroel5.out',status='unknown')
      read(5,*) np
      do i=1,n
         read(5,*) dum,f(i)
      enddo
      c0=0.
      c1=0.
      s1=0.
      c2=0.
      s2=0.
      do i=1,n
         c0=c0+f(i)
         s1=s1+f(i)*sin(6.2832*(i-1)/n)
         c1=c1+f(i)*cos(6.2832*(i-1)/n)
         s2=s2+f(i)*sin(6.2832*(i-1)/m)
         c2=c2+f(i)*cos(6.2832*(i-1)/m)
       enddo
       c0=c0/n
       s1=s1/m
       c1=c1/m
       s2=s2/m
       c2=c2/m
       write(9,*)
       write(9,*) 'Fourier Coefficients'
       write(9,*) 'c0 s1 c1 s2 c2'
       write(9,1000) c0,s1,c1,s2,c2
C
C      FORMAT STATEMENTS
C
1000   format(5(1x,f7.4))
       end
```

4.4 Flutter Divergence

```
      PROGRAM DIVERGENCE
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     STATIC DIVERGENCE CALCULATION
C
C     Input:
C
C     AERODYNAMIC INTEGRALS
C
C     I1 1.0          I1 AERODYNAMIC INTEGRAL
C     I2 0.6779       I2 AERODYNAMIC INTEGRAL
C     I3 0.5          I3 AERODYNAMIC INTEGRAL
C     I4 0.783        I4 AERODYNAMIC INTEGRAL
C     I5 0.6366       I5 AERODYNAMIC INTEGRAL
C
C     LINEAR AERODYNAMIC COEFFICIENTS LIFT/MOMENT
C
C     AOL  5.73       AOL  ONERA LINEAR COEFFICIENT FOR LIFT
C     B1   8.6        NON-LINEAR SLOPE OF THE DEVIATION FROM LINEAR FORCE CURVE  - LIFT
C     B2   0.4        NON-LINEAR SLOPE OF THE DEVIATION FROM LINEAR FORCE CURVE  - MOMENT
C
C     STRUCTURE - MATERIALS
C
C     M   0.283       TOTAL MASS PER UNIT LENGTH (kg/m)
C     IA  0.343E-3    MASS MOMENT OF INERTIA PER UNIT LENGTH (kg m)
C     WH  27.02       UNCOUPLED BENDING FREQUENCY (rad/s)
C     WA  154.6       UNCOUPLED TORSIONAL FREQUENCY  (rad/s)
C     RHO 1.23        WING MATERIAL DENSITY FREE STREAM DENSITY (kg/m^3)
C
C     GEOMETRY
C
C     C   0.140       CHORD LENGTH (m)
C     L   0.559       WING SPAN (m)
C     ALPHA1D  10.    STALL ANGLE DEGREES
C     THETA0D  0.     ROOT ANGLE DEGREES
C
C     INITIAL GUESS
C
C     x(1)=Q10        MEAN COMPONENT FIRST MODE (BENDING)
C     x(2)=Q20        MEAN COMPONENT SECOND MODE (TORSION)
C
C     MODE SHAPE
C
C     PHIA 0.84460    BEAM TORSION MODE SHAPE
C
C     CONVERGENCE
C
C     NITS  25        MAX NUMBER OF NEWTON-RAPHSON ITERATIONS
C     TOLER 1.E-4     TOLERANCE FOR EACH OF THE 6 UNKNOWNS
C
C     PARAMETERIC STUDY
C
C     Q2S 0.1         SINE COMPONENT OF THE SECOND MODAL FORCE
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      PARAMETER (NN=2)
      INTEGER N,NP,I,T,J,INDX(NN),NITS
      REAL*8
     1      X(1:NN),FVEC(1:NN),FJAC(1:NN,1:NN),DELTA(NN),
     2      ALPHA0,PHI1,TOLER
      REAL  Y(NN,NN),D,THETA0,THETA0D,KAT
C
      REAL
     1      I1,I2,I3,I4,I5,AOL,B1,B2,PHIA,AVIBD,AVIB,
     2      M,IA,RHO,C,L,WH,WA,ALPHA1D,ALPHA1,B,PI,OM,MU,RA
      COMMON/USER/
     1      I1,I2,I3,I4,I5,AOL,B1,B2,PHIA,AVIBD,AVIB,
     2      M,IA,RHO,C,L,WH,WA,ALPHA1D,ALPHA1,B,PI,OM,MU,RA
C
C     READ DATA FROM THE INPUT FILE
C
      OPEN (10,FILE='aeroel6.in',STATUS='OLD')
      READ (10,120) I1,I2,I3,I4,I5,AOL,B1,B2
      READ (10,120) M,IA,RHO,C,L,WH,WA,ALPHA1D,THETA0D
```

```
      READ (10,120) X(1),X(2),PHIA,AVIBD
      READ (10,*)   NITS
      READ (10,120) TOLER,KAT
C
C     CREATE OUTPUT FILE
C
      OPEN  (9,FILE='aeroel6.out',STATUS='UNKNOWN')
      WRITE (9,*)
      WRITE (9,*) '# Calculation of Flutter Divergence'
      WRITE (9,*)
      WRITE (9,105)
     1      'Theta-R','K2','Q10','Q20','AOA(deg)','V(m/2)'
C
C SETUP PARAMETERS
C
      THETA0=DEG2RAD(THETA0D)
      AVIB  =DEG2RAD(AVIBD)
      N=2
      NP=2
      ALPHA1=DEG2RAD(ALPHA1D)
      B=0.5*C
      PI=3.14159
      OM=WH/WA
      MU=(M/L)/(PI*RHO*B*B)
      RA=SQRT(IA/M)/B
C
C     NEWTON STEP
C
      T=0
      DO I=1,N
         DELTA(I)=5.
      ENDDO
20    CONTINUE
      print *, 'START',T
      IF (DABS(DELTA(1)).LE.TOLER.AND.
     2    DABS(DELTA(2)).LE.TOLER) GOTO 90
      T=T+1
C
C     CALCULATE THE FUNCTIONS USING THE INITIAL VALUES FOR THE
C     STATE VECTOR X
C
      CALL FUNCV(KAT,THETA0,N,X,FVEC)
C
C     CALCULATE THE JACOBIAN MATRIX
C
      CALL FDJAC(KAT,THETA0,N,X,FVEC,NP,FJAC)
C
      DO I=1,N
         DO J=1,N
            Y(I,J)=0.
         ENDDO
         Y(I,I)=1.
      ENDDO
C
C     CALCULATE THE INVERSE OF THE JACOBIAN MATRIX
C
      CALL LUDCMP (FJAC,N,NP,INDX,D)
      DO J=1,N
         CALL LUBKSB (FJAC,N,NP,INDX,Y(1,J))
      ENDDO
C
C     CALCULATE DELTA
C
      DO J=1,N
         DELTA(J)=0.
         DO I=1,N
            DELTA(J)=DELTA(J)+Y(J,I)*FVEC(I)
         ENDDO
      ENDDO
C
C     CALCULATE X AT THE NEXT ITERATION
C
      DO I=1,N
         X(I)=X(I)-DELTA(I)
      ENDDO
C
      IF (T.LT.NITS) GOTO 20
C
```

```
C     PRINTOUT
C
      CONTINUE
      print *, T
      IF (T.GT.NITS) THEN
         WRITE (*,*) 'DIVERGENCE'
         STOP
      ENDIF
C
C     CALCULATE FLUTTER VELOCITY AND FREQUENCY
C
90    V=WA*B/KAT
      AOA=THETA0+0.5*PHIA*X(2)
      WRITE (9,110)THETA0,KAT,X(1),X(2),RAD2DEG(AOA),V
C
C     NEXT KAT ENTRY
C
      T=0
      DO I=1,N
         X(I)=X(I)+0.0001
      ENDDO
      DO I=1,N
         DELTA(I)=5.
      ENDDO
      READ (10,*) KAT
      IF (KAT.LT.999.0) GOTO 20
C
      STOP
C
C     FORMAT STATEMENTS
C
105   FORMAT('#',11(1X,A11))
110   FORMAT(11(1X,F11.4))
120   FORMAT(1X,F10.5)
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE FDJAC (KAT,THETA0,N,X,FVEC,NP,DF)
C
C     THE JACOBIAN MATRIX IS CALCULATED INCREASING
C     THE STATE VECTOR BY A SMALL NUMBER
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      PARAMETER (NMAX=40)
      INTEGER N,NP,NMAX
      REAL*8 DF(NP,NP),FVEC(N),X(N),EPS,F(NMAX)
      INTEGER I,J
      REAL H,TEMP,THETA0,KAT
      EPS=1.E-4
      DO J=1,N
         TEMP=X(J)
         H=EPS*ABS(TEMP)
         IF (H.EQ.0.) H=EPS
         X(J)=TEMP+H
         H=X(J)-TEMP
         CALL FUNCV (KAT,THETA0,N,X,F)
         X(J)=TEMP
         DO I=1,N
            DF(I,J)=(F(I)-FVEC(I))/H
         ENDDO
      ENDDO
C
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE FUNCV (KAT,THETA0,N,X,P)
C
C     THE AEROELASTIC MODEL IS USED HERE TO FORMULATE THE EQUATIONS
C     WHICH ARE TO BE SOLVED (P). THE VARIABLES CAN BE EASILY IDENTIFIED
C     WITH THE NAME OF THE VARIABLES IN THE DISSERTATION.
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      INTEGER N
      REAL Q2S,THETA0,KAT
      REAL*8 X(N),P(N),CL20,CM20,ALPHA0,BOL,PENT,PHI1
C
      REAL
     1      I1,I2,I3,I4,I5,AOL,B1,B2,PHIA,AVIBD,AVIB,
     2      M,IA,RHO,C,L,WH,WA,ALPHA1D,ALPHA1,B,PI,OM,MU,RA
      COMMON/USER/
     1      I1,I2,I3,I4,I5,AOL,B1,B2,PHIA,AVIBD,AVIB,
```

```
     2      M,IA,RHO,C,L,WH,WA,ALPHA1D,ALPHA1,B,PI,OM,MU,RA
C
C      ANGLE OF ATTACK IN STRUCTURAL CO-ORDINATES
C
       ALPHA0=THETA0+.5*PHIA*X(2)
C
C      SINGLE BREAK-POINT
C
       PENT=(ALPHA1-ALPHA0)/AVIB
       IF (PENT.GT.1.) THEN
          PHI1=0.5*PI
       ELSE IF (PENT.LT.-1.) THEN
          PHI1=-0.5*PI
       ELSE
          PHI1=DASIN(PENT)
       ENDIF
C
C      NON-LINEAR PART - L REFERS TO LIFT
C
       CL20=-(B1*AVIB/PI)*(-PENT*(1.571-PHI1 )+DCOS(PHI1 ))
C
C      NON-LINEAR PART - M REFERS TO MOMENT
C
       CM20=-(B2*AVIB/PI)*(-PENT*(1.571-PHI1 )+DCOS(PHI1 ))
C
C      EQUATIONS TO SOLVE
C
       B0L=0.25*A0L*X(2)
       P(1)=MU*PI*I1*OM*OM*KAT*X(1)-I4*AOL*THETA0-I2*BOL-I4*CL20
C
       P(2)=MU*(PI/4.)*RA*RA*I3*kat*kat*X(2)-.25*I5*AOL*THETA0
      1     -0.25*I3*BOL-0.25*I5*CL20-I5*CM20
C
       RETURN
       END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
       SUBROUTINE LUDCMP (A,N,NP,INDX,D)
C
C      LU MATRIX DECOMPOSITION
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
       PARAMETER (NMAX=100,TINY=1.E-20)
       DOUBLE PRECISION A(NP,NP),VV(NMAX)
       DIMENSION INDX(N)
C
       D=1.
       DO I=1,N
          AAMAX=0.
          DO J=1,N
             IF (DABS(A(I,J)).GT.AAMAX) AAMAX=DABS(A(I,J))
          ENDDO
          IF (AAMAX.EQ.0.) PAUSE 'SINGULAR MATRIX IN LUDCMP'
          VV(I)=1./AAMAX
       ENDDO
       DO J=1,N
          DO I=1,J-1
             SUM=A(I,J)
             DO K=1,I-1
                SUM=SUM-A(I,K)*A(K,J)
             ENDDO
             A(I,J)=SUM
          ENDDO
          AAMAX=0.
          DO I=J,N
             SUM=A(I,J)
             DO K=1,J-1
                SUM=SUM-A(I,K)*A(K,J)
             ENDDO
             A(I,J)=SUM
             DUM=VV(I)*ABS(SUM)
             IF (DUM.GE.AAMAX) THEN
                IMAX=I
                AAMAX=DUM
             ENDIF
          ENDDO
          IF (J.NE.IMAX) THEN
             DO K=1,N
                DUM=A(IMAX,K)
                A(IMAX,K)=A(J,K)
```

```
               A(J,K)=DUM
            ENDDO
            D=-D
            VV(IMAX)=VV(J)
         ENDIF
         INDX(J)=IMAX
         IF (A(J,J).EQ.0.) A(J,J)=TINY
         IF (J.NE.N) THEN
            DUM=1./A(J,J)
            DO I=J+1,N
               A(I,J)=A(I,J)*DUM
            ENDDO
         ENDIF
      ENDDO
C
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE LUBKSB (A,N,NP,INDX,B)
C
C     BACK SUBSTITUTION
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      DIMENSION INDX(N),B(N)
      DOUBLE PRECISION A(NP,NP)
C
      II=0
      DO I=1,N
         LL=INDX(I)
         SUM=B(LL)
         B(LL)=B(I)
         IF (II.NE.0) THEN
            DO J=II,I-1
               SUM=SUM-A(I,J)*B(J)
            ENDDO
         ELSE IF (SUM.NE.0.) THEN
            II=I
         ENDIF
         B(I)=SUM
      ENDDO
      DO I=N,1,-1
         SUM=B(I)
         DO J=I+1,N
            SUM=SUM-A(I,J)*B(J)
         ENDDO
         B(I)=SUM/A(I,I)
      ENDDO
C
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION DEG2RAD(ANGLE)
C
C     CONVERT DEG-->RAD
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL ANGLE
C
      DEG2RAD=ANGLE*ATAN(1.)/45.
C
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION RAD2DEG(ANGLE)
C
C     CONVERT RAD-->DEG
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL ANGLE
C
      RAD2DEG=ANGLE*45./ATAN(1.)
C
      RETURN
      END
```